T.C.

DİCLE ÜNİVERSİTESİ

DİYARBAKIR TEKNİK BİLİMLER MESLEK YÜKSEKOKULU

BİLGİSAYAR TEKNOLOJİLERİ BÖLÜMÜ

ÖĞRENCİ BİLGİ SİSTEM OTOMASYONU

Projeyi Hazırlayan

Harun ÜNGER

Projeyi Yöneten

Dr. Öğr. Üyesi İsa ATAŞ

Bilgisayar Teknolojileri Bölümü

Sistem Analizi ve Tasarımı Projesi

Haziran 2023

DİYARBAKIR

            Bu çalışma, Dicle Üniversitesi Diyarbakır Teknik Bilimler Meslek Yüksekokulu **Bilgisayar Teknolojileri Bölümü Bilgisayar Programcılığı Programı** IV. dönem “Sistem Analizi ve  Tasarım” dersinin proje  raporu olarak kabul edilmiştir.

                                                                                               ....../…… /2023

            Projeyi Hazırlayan:

1. Adı: Harun ÜNGER
2. Numarası:
3. Puanı:……..

            ONAY:

            Bu proje çalışması onaylanmış ve ….. öğrencinin Sistem Analizi ve Tasarımı dersinden geçmesine/kalmasına karar verilmiştir.

                                                                                          ....../……/2023

                                                                                   Proje Danışmanı

Dr. Öğr. Üyesi İsa ATAŞ

İmza

ÖZET

SİSTEM ANALİZİ VE TASARIMI DERSİ DÖNEM PROJESİ

ÖĞRENCİ BİLGİ SİSTEM OTOMASYONU GELİŞTİRİLMESİ

Projeyi Hazırlayan

Harun ÜNGER

Dicle Üniversitesi

Diyarbakır Teknik Bilimler Meslek Yüksekokulu

Bilgisayar Teknolojileri Bölümü

2023, Sayfa:67

Bu çalışma, C#, MSSQL, Visual Studio ve Windows Form kullanılarak küçük çaplı bir öğrenci otomasyon sistemi tasarımı ve geliştirmesi üzerine odaklanmaktadır. Projenin amacı, üniversite öğrencilerine yönelik bir otomasyon sistemi oluşturarak akademik süreçlerini daha etkin bir şekilde yönetmelerini sağlamaktır. İlk olarak, projenin gereksinimleri ve kullanıcı ihtiyaçları analiz edildi ve tasarım aşamasına geçildi. Ardından, C# programlama dili ve Visual Studio geliştirme ortamı kullanılarak form ekranları oluşturuldu ve işlevsel hale getirildi. MSSQL veri tabanı yönetim sistemi kullanılarak veri tabanı tabloları oluşturuldu ve ilişkilendirme yapıldı. Son olarak, sistem test edilerek hatalar giderildi. Çalışma sonucunda elde edilen bulgular, tasarlanan öğrenci otomasyon sisteminin başarılı bir şekilde işlevsel hale getirildiğini ve öğrencilerin akademik süreçlerini etkin bir şekilde yönetmelerine yardımcı olduğunu gösterdi. Öğrenciler, kolay kullanılabilir arayüz sayesinde ders seçimi yapabilir ve notlarını görüntüleyebilirler. Öğrencilerin akademik süreçlerini etkin bir şekilde yönetmeleri, daha verimli bir eğitim deneyimi yaşamalarını sağlar. Bu proje, C#, MSSQL, Visual Studio ve Windows Form gibi teknolojilerin kullanımının bu tür projelerde kolaylık sağladığını ortaya koymaktadır.

Anahtar Kelimeler: Öğrenci Otomasyon Sistemi, C#, MSSQL, Visual Studio, Windows Form.

**İÇİNDEKİLER**

[1. GİRİŞ 9](#_Toc137760303)

[2. OTOMASYON 10](#_Toc137760304)

[2.1 Otomasyon Nedir? 10](#_Toc137760305)

[2.2 Öğrenci Otomasyonu nedir? 11](#_Toc137760306)

[2.3 C# Nedir? 11](#_Toc137760307)

[2.4 C# Tarihçesi 12](#_Toc137760308)

[2.5 Nesne Tabanlı Programlama Nedir? 12](#_Toc137760309)

[2.6 Windows Form ve Visual Studio Nedir ve Nerelerde Kullanılır? 13](#_Toc137760310)

[2.7 Görsel Arayüz Tasarımının Rolü 13](#_Toc137760311)

[2.8 Veri tabanı Nedir? 14](#_Toc137760312)

[2.9 Veri tabanı Türleri Nelerdir? 14](#_Toc137760313)

[2.10 MSSQL Nedir ve Nerelerde Kullanılır? 15](#_Toc137760314)

[2.11 C# ve MSSQL Kullanarak Otomasyon Yapımı 15](#_Toc137760315)

[3. PROJE ANALİZİ: C# İLE ÖĞRENCİ OTOMASYONU GELİŞTİRME 16](#_Toc137760316)

[3.1 Proje Gereksinimleri 16](#_Toc137760317)

[3.2 Proje Geliştirme Yaklaşımı ve Teknolojiler: 16](#_Toc137760318)

[3.3 Kullanılan C# Kütüphaneleri ve Projeye Eklenen Özellikler: 17](#_Toc137760319)

[3.4 Proje Geliştirme Süreci ve Karşılaşılan Zorluklar 17](#_Toc137760320)

[4. DİZAYN 19](#_Toc137760321)

[4.1 User Case Diagram 19](#_Toc137760322)

[4.2 Veri tabanı er diyagramları şema tasarımları ve tablolar 20](#_Toc137760323)

[4.3 Tablolar 20](#_Toc137760324)

[4.3.1 BolumTable 22](#_Toc137760325)

[4.3.2 DersTable 22](#_Toc137760326)

[4.3.3 MemurTable 23](#_Toc137760327)

[4.3.4 NotTable 23](#_Toc137760328)

[4.3.5 OgrenciTable 24](#_Toc137760329)

[4.3.6 OgretmenDersTable 24](#_Toc137760330)

[4.3.7 OgretmenTable 25](#_Toc137760331)

[4.3.8 UserTable 25](#_Toc137760332)

[4.3.9 OgrenciDersTable 26](#_Toc137760333)

[5. TASARIM 27](#_Toc137760334)

[5.1 Giriş Paneli 27](#_Toc137760335)

[5.2 Memur Paneli 27](#_Toc137760336)

[5.3 Kişi Ekleme Paneli 28](#_Toc137760337)

[5.4 Kişi Güncelleme Paneli 28](#_Toc137760338)

[5.5 Öğretmen Paneli 29](#_Toc137760339)

[5.6 Ders Onay Paneli 30](#_Toc137760340)

[5.7 Not Ekleme Paneli 30](#_Toc137760341)

[5.8 Öğrenci Paneli 31](#_Toc137760342)

[5.9 Ders Seçimi 31](#_Toc137760343)

[5.10 Not Görüntüle 32](#_Toc137760344)

[6. KODLAMA 33](#_Toc137760345)

[6.1 Kullanılan kütüphaneler 33](#_Toc137760346)

[6.2 girisPanel.cs 34](#_Toc137760347)

[6.3 memurPanel.cs 37](#_Toc137760348)

[6.4 kisiEklePanel.cs 38](#_Toc137760349)

[6.5 kisiGuncellePanel.cs 43](#_Toc137760350)

[6.6 ogretmenPanel.cs 48](#_Toc137760351)

[6.7 notEklePanel.cs 50](#_Toc137760352)

[6.8 dersOnayPanel.cs 53](#_Toc137760353)

[6.9 ogrenciPanel.cs 55](#_Toc137760354)

[6.10 notGoruntulePanel.cs 59](#_Toc137760355)

[6.11 dersSecimPanel.cs 61](#_Toc137760356)

[6.12 User.cs 64](#_Toc137760357)

[7. SONSÖZ 65](#_Toc137760358)

[8 . KAYNAKÇA 66](#_Toc137760359)

Öğrencinin;

Adı Soyadı: Harun Ünger

Email Adresi: unger.aaron.dev@gmail.com

1. GİRİŞ

Bu tez çalışması, C#, MSSQL, Visual Studio ve Windows Form gibi teknolojileri kullanarak üniversite öğrencileri için küçük çaplı bir öğrenci otomasyon sistemi tasarlanmasını amaçlamaktadır. Günümüzde teknolojinin hızla gelişmesiyle birlikte, eğitim alanında da bilgisayar tabanlı sistemlerin kullanımı büyük önem kazanmıştır. Öğrenci otomasyon sistemleri, öğrenci yönetimi süreçlerinin etkin bir şekilde yürütülmesine yardımcı olmak amacıyla geliştirilmektedir.

Çalışmanın amacı, üniversite öğrencilerinin akademik süreçlerini kolaylaştırmak ve yönetmek için kullanışlı bir otomasyon sistemi tasarlamaktır. Geliştirilecek sistem, ders seçimi, not girişi, öğrenci ve öğretmen bilgilerinin yönetimi gibi işlevleri içermektedir. Böylece, öğrencilerin ders seçimi yapma ve notlarını görüntüleme gibi işlemleri daha verimli bir şekilde gerçekleştirmeleri amaçlanmaktadır.

Tezin konusu, C#, MSSQL, Visual Studio ve Windows Form teknolojilerini kullanarak öğrenci otomasyon sistemi tasarımı ve geliştirme sürecidir. Bu bağlamda, sistemde kullanılan form ekranları ve veri tabanı tabloları dikkate alınarak sistemin bütünlüğü sağlanmaktadır. Tasarlanan sistem, kullanıcı dostu bir arayüz ve güvenilir bir veri tabanı yapısıyla öğrenci yönetimi süreçlerine katkı sağlamaktadır.

Bu çalışmanın önemi, eğitim sektöründe bilgisayar tabanlı sistemlerin kullanımının artmasıyla birlikte öğrenci otomasyon sistemlerinin önemli bir ihtiyaç haline gelmesidir. Tasarlanan sistem, öğrencilerin akademik süreçlerini etkin bir şekilde yönetmelerine yardımcı olacak ve veri güvenliği sağlayarak doğru bilgilerin güncellenmesini sağlayacaktır. Ayrıca, bu çalışma, C#, MSSQL, Visual Studio ve Windows Form gibi teknolojilerin kullanımının kolaylıklarını ve avantajlarını vurgulayarak bu teknolojilere yönelik farkındalığı artırmayı amaçlamaktadır.

Sonuç olarak, bu tez çalışması, C#, MSSQL, Visual Studio ve Windows Form teknolojilerini kullanarak üniversite öğrencileri için küçük çaplı bir öğrenci otomasyon sistemi tasarlamayı amaçlamaktadır. Tasarlanacak sistem, öğrenci yönetimi süreçlerini kolaylaştıracak, verimliliği artıracak ve doğru bilgilerin güncellenmesini sağlayacaktır. Bu çalışma, eğitim sektöründe bilgisayar tabanlı sistemlerin etkin bir şekilde kullanılmasını desteklemekte ve öğrenci yönetimi süreçlerine katkı sağlamaktadır.

1. OTOMASYON

Günümüzde otomasyon, birçok sektörde büyük önem kazanmıştır. İnsan emeği yerine makinelerin ve bilgisayarların kullanılmasıyla iş süreçleri daha verimli hale getirilirken, hata oranları da azaltılmaktadır. Bu nedenle, otomasyon sistemlerinin geliştirilmesi ve kullanımı büyük bir ilgi alanı olmuştur.

Bu tezde, özellikle bir öğrenci yönetim sistemi otomasyonu üzerinde odaklanılmaktadır. Geleneksel öğrenci yönetim süreçlerinde, birçok işlem manuel olarak gerçekleştirilmekte ve zaman alıcı olabilmektedir. Ancak, otomasyon sayesinde bu süreçlerin daha hızlı, daha doğru ve daha etkin bir şekilde yönetilmesi hedeflenmektedir.

Tez kapsamında, C# programlama dili, MSSQL veri tabanı, Visual Studio ve Windows Form teknolojileri kullanılarak bir öğrenci yönetim sistemi otomasyonu geiştirildi. Bu otomasyon sistemi, öğrenci kaydı, not girişi, ders seçimi, öğretmen ve öğrenci bilgilerinin yönetimi gibi önemli işlemleri içermektedir.

Otomasyonun sağladığı verimlilik, doğruluk ve kullanıcı kolaylığı gibi unsurların vurgulanmasıyla, bu çalışmanın öğrenci yönetimi alanındaki bir katkı sunacağı düşünüldü.

## 2.1 Otomasyon Nedir?

Otomasyon, bir sürecin veya işin, insan müdahalesi olmaksızın, makinelerin veya bilgisayarların kontrolünde gerçekleştirilmesidir. Otomasyon, tekrarlanabilir ve rutin görevleri otomatikleştirerek zaman ve emek tasarrufu sağlar. İş süreçlerindeki hataları azaltır, verimliliği artırır ve işlerin daha hızlı bir şekilde tamamlanmasını sağlar.

Otomasyon, birçok sektörde kullanılmakta ve çeşitli alanlarda büyük faydalar sağlamaktadır. Örneğin, endüstriyel otomasyon, fabrikalardaki üretim süreçlerinin robotlar ve otomatik makineler aracılığıyla gerçekleştirilmesini ifade eder. Bu sayede üretim hızı artar, kalite kontrolü iyileşir ve işçi güvenliği sağlanır.

Otomasyonun sağladığı verimlilik ve doğruluk, insan hatası riskini azaltır ve iş süreçlerinin daha güvenilir bir şekilde yürütülmesini sağlar. Bunun yanı sıra, otomasyon sistemleri veri analizi ve raporlama gibi işlemleri de gerçekleştirerek yöneticilere daha iyi kararlar alma imkanı sunar.

Sonuç olarak, otomasyon iş süreçlerinde verimlilik, doğruluk ve zaman tasarrufu sağlayan bir yaklaşımdır. Günümüzde birçok sektörde yaygın olarak kullanılan otomasyon sistemleri, iş süreçlerini iyileştirerek kurumların rekabetçi avantaj elde etmelerini sağlar. Bu nedenle, otomasyonun faydalarını anlamak ve etkin bir şekilde kullanmak, çağın gereksinimlerine uyum sağlamak açısından önemlidir.

## 2.2 Öğrenci Otomasyonu nedir?

Öğrenci otomasyonu, bir eğitim kurumunda öğrencilerle ilgili süreçlerin ve verilerin yönetimini kolaylaştıran bir sistemdir. Bu sistem, öğrencilerin kayıt işlemlerinden ders seçimine, not girişinden öğretmen kayıt işlemine kadar çeşitli işlemleri otomatikleştirir ve verimli bir şekilde yönetir.

Öğrenci otomasyonu, öğrenci bilgilerinin merkezi bir veri tabanında saklandığı ve erişilebilir olduğu bir yapıya sahiptir. Bu sayede, öğrencilerin kişisel bilgileri, akademik takipleri, ders programları, notları ve diğer ilgili veriler güvenli bir şekilde saklanır ve kolayca erişilebilir hale gelir.

Ayrıca, öğrenci otomasyonu öğrenci ve akademik personel arasındaki iletişimi kolaylaştırır. Öğrenciler, otomasyon sistemi üzerinden ders kaydı yapabilir, ders programlarını takip edebilir ve notlarını görüntüleyebilir. Öğretmenler ise otomasyon sistemi aracılığıyla öğrenci notlarını girebilir, devamsızlık takibi yapabilir ve öğrencilerle iletişim halinde olabilir. Bu sayede, öğrenci otomasyonu, eğitim kurumlarının verimliliğini artırır, kaynakları daha etkin bir şekilde yönetmelerini sağlar ve öğrencilerin akademik başarılarını destekler.

Öğrenci otomasyonunun sağladığı veri güvenliği, doğruluk ve kolay erişilebilirlik avantajları, eğitim kurumlarında yönetim süreçlerini kolaylaştırır ve iş yükünü azaltır. Ayrıca, öğrenci otomasyonu sayesinde öğrenci performansı takip edilebilir, analiz edilebilir ve gerekli müdahaleler yapılabilir. Bu da öğrencilerin gelişimini destekleyen bir eğitim ortamının oluşturulmasına katkı sağlar.

Sonuç olarak, öğrenci otomasyonu, eğitim kurumlarında öğrenci yönetim süreçlerinin etkin bir şekilde yürütülmesini sağlayan bir sistemdir. Veri güvenliği, doğruluk ve kolay erişilebilirlik gibi avantajlarıyla öğrenci otomasyonu, eğitim kalitesini artırırken yönetim süreçlerini de optimize eder. Bu nedenle, eğitim kurumlarının modernize olması ve daha etkin hizmet sunması için öğrenci otomasyonunun kullanılması önemli bir adımdır.

## 2.3 C# Nedir?

C#, Microsoft tarafından geliştirilen ve .NET Framework üzerinde çalışan bir nesne yönelimli programlama dilidir. C#, modern, güvenilir ve yüksek performanslı uygulamalar geliştirmek için kullanılan popüler bir dil olarak kabul edilir. Nesne tabanlı bir yapıya sahip olan C#, kullanıcı dostu ve anlaşılması kolay bir sözdizimine sahiptir.

C#, zengin bir kütüphane setiyle birlikte gelir ve geniş bir uygulama yelpazesine hitap eder. Windows tabanlı masaüstü uygulamaları, web uygulamaları, mobil uygulamalar, oyunlar ve hatta gömülü sistemler gibi çeşitli alanlarda kullanılabilir. C# dilinin sağladığı özellikler arasında güçlü veri işleme yetenekleri, çoklu programlama paradigmasını destekleyen esneklik ve hata yönetimi mekanizmaları bulunur. C#, geliştiricilere zengin bir geliştirme ortamı sunar ve geniş bir topluluk tarafından desteklenir.

## 2.4 C# Tarihçesi

C#, 2000 yılında Microsoft tarafından geliştirilmeye başlanmıştır. İlk olarak, Microsoft'un .NET Framework'u için tasarlanmıştır ve .NET dilleri arasında en popülerlerinden biri olmuştur. C#, hızla gelişmiş ve günümüzde geniş bir kullanıcı ve geliştirici kitlesi tarafından benimsenmiştir.

C#'ın tarihçesi, dilin sürekli geliştirilerek yeni özelliklerin eklenmesiyle şekillenmiştir. Örneğin, C# 2.0 ile jenerikler ve anonim metodlar gibi önemli özellikler tanıtılmıştır. C# 3.0 ile birlikte LINQ (Language Integrated Query) ve lambda ifadeleri gibi dilin daha da güçlenmesini sağlayan özellikler eklenmiştir. C# 5.0, 6.0, 7.0, 8.0 ve 9.0 sürümleri de sırasıyla yeni özellikler, iyileştirmeler ve dilin daha da gelişmesiyle ilgili yenilikler sunmuştur.

## 2.5 Nesne Tabanlı Programlama Nedir?

Nesne tabanlı programlama (NTP), bir programı nesnelerin oluşturduğu ve birbirleriyle etkileşime geçtiği bir yaklaşımla tasarlama ve geliştirme yöntemidir. C#, Java, C++ gibi programlama dilleri, nesne tabanlı programlamanın temel özelliklerini destekler.

Nesne tabanlı programlamada, gerçek dünyadaki varlıkları (nesneleri) programlamada temel birim olarak kullanırız. Her nesne, verileri (öznitelikler) ve bu veriler üzerinde işlemler yapabilen fonksiyonları (metodlar) içerir. Nesneler, bir sınıfın (class) örnekleridir ve bir sınıf, bir nesnenin özelliklerini ve davranışlarını tanımlayan bir şablondur.

Nesne tabanlı programlama, programların daha modüler, esnek ve yeniden kullanılabilir olmasını sağlar. Kodun daha iyi organize edilmesini ve bakımının kolaylaşmasını sağlar. Ayrıca, kalıtım, polimorfizm ve kapsülleme gibi kavramlarla birlikte programlamadaki karmaşıklığı azaltır ve kodun daha anlaşılır olmasını sağlar. Nesne tabanlı programlama, büyük ve karmaşık projelerin geliştirilmesinde etkin bir yaklaşım olarak kullanılır.

## 2.6 Windows Form ve Visual Studio Nedir ve Nerelerde Kullanılır?

Windows Form, C# ile Windows tabanlı uygulamalar geliştirmek için kullanılan bir arayüz teknolojisidir. Windows Form, kullanıcı dostu ve interaktif arayüzler oluşturmak için kullanılır. Windows Form uygulamaları, düğmeler, metin kutuları, listeler, menüler gibi çeşitli kontrolleri kullanarak kullanıcıyla etkileşim sağlar.

Visual Studio, Microsoft tarafından geliştirilen bir entegre geliştirme ortamıdır (IDE). C# kodunun yazılması, hata ayıklanması, derlenmesi ve uygulamanın oluşturulması için kullanılır. Visual Studio, Windows Form uygulamalarının hızlı ve kolay bir şekilde tasarlanmasını sağlar. Görsel bir arayüzle kod yazma imkanı sunar ve otomatik tamamlama, hata kontrolü gibi geliştirme sürecini destekleyen araçlar içerir.

Windows Form ve Visual Studio, masaüstü uygulamalarının geliştirilmesinde yaygın olarak kullanılır. Örneğin, ofis uygulamaları, veri tabanı yönetim araçları, grafiksel kullanıcı arayüzleri gibi çeşitli alanlarda Windows Form ve Visual Studio kullanılabilir. Kullanıcı dostu ve görsel olarak zengin arayüzler oluşturmak ve Windows platformuna yönelik uygulamalar geliştirmek için tercih edilirler.

## 2.7 Görsel Arayüz Tasarımının Rolü

Projede Windows Form kullanarak görsel bir arayüz tasarımı oluşturmak, işleri kolaylaştıran önemli bir faktördü. Windows Form, kullanıcıların projeyi daha rahat anlamalarını ve kullanmalarını sağlayan bir grafiksel kullanıcı arayüzü (GUI) sağlar. İşte Windows Form'un görsel arayüz tasarımında sağladığı kolaylıklar:

1. Kullanılabilirlik: Windows Form, kullanıcı dostu bir arayüz oluşturma konusunda etkili bir yol sunar. Görsel nesneleri (textbox, combobox, label, button vb.) sürükle-bırak yöntemiyle yerleştirebilir ve düzenleyebilirsiniz. Bu, kullanıcıların projeyi daha hızlı ve kolay anlamalarına yardımcı olur.
2. Araçlar ve Kontroller: Windows Form, birçok kullanışlı araç ve kontrol sunar. Bu araçlar, kullanıcı girişi almak, verileri göstermek, butonlarla işlevleri tetiklemek ve daha fazlasını yapmak için kullanılabilir. Örneğin, textboxlar kullanıcıdan veri girişi almak için kullanılabilir, comboboxlar seçenekleri listelemek için kullanılabilir ve butonlar belirli işlevleri gerçekleştirmek için kullanılabilir. Bu araçlar, projenin işlevselliğini artırır ve kullanıcıların etkileşimini kolaylaştırır.
3. Görsel Düzenleme: Windows Form, görsel düzenlemeyi kolaylaştırır. Kontrolleri ve nesneleri sürükle-bırak yöntemiyle yerleştirebilir, boyutlandırabilir ve hizalayabilirsiniz. Böylece, projenin estetik açıdan daha çekici ve kullanıcı dostu olmasını sağlayabilirsiniz. Renkler, fontlar, arka plan resimleri ve diğer özelliklerle tasarımı özelleştirebilirsiniz.
4. Olay Yönetimi: Windows Form, kullanıcının etkileşimlerini yönetmek için olay tabanlı bir model kullanır. Örneğin, bir butona tıklandığında belirli bir işlevin çalıştırılmasını sağlayabilirsiniz. Bu şekilde, kullanıcının belirli bir olaya yanıt vermesini ve projenin istenen şekilde çalışmasını sağlayabilirsiniz.

Sonuç olarak, Windows Form kullanarak görsel bir arayüz tasarlamak, projenin kullanılabilirliğini artırır ve işleri kolaylaştırır. Kullanıcı dostu bir arayüz sağlamak, projenin daha anlaşılır ve kullanılabilir olmasını sağlar. Ayrıca, görsel düzenleme ve olay yönetimi gibi özellikler sayesinde projeyi istenen şekilde özelleştirebilir ve kullanıcı deneyimini iyileştirebilirsiniz.

## 2.8 Veri tabanı Nedir?

Veri tabanı, verilerin yapılandırılmış bir şekilde saklandığı bir depolama alanıdır. Bir veri tabanı, veri tabloları, ilişkiler ve sorgular aracılığıyla verilerin etkili bir şekilde yönetilmesini sağlar. Veritabanları, bilgileri düzenlemek, depolamak, güncellemek ve sorgulamak için kullanılır.

Veritabanları, verilerin tutarlılığını ve bütünlüğünü sağlamak için çeşitli kısıtlamalar ve ilişkiler kullanır. Veri tabloları, verilerin yapılandırıldığı ve ilgili veri gruplarının bir araya getirildiği tablolardır. Tablolar, sütunlar (alanlar) ve satırlar (kayıtlar) olarak düzenlenir. Veri tabanı yönetim sistemi (DBMS), veri tabanıyla etkileşimde bulunmayı sağlar ve verilerin güvenli ve verimli bir şekilde yönetilmesini sağlar.

## 2.9 Veri tabanı Türleri Nelerdir?

Veritabanları, farklı yapıları ve kullanım amaçlarına göre farklı türlerde olabilir. İşte bazı yaygın veri tabanı türleri:

İlişkisel Veri tabanı: İlişkisel veritabanları, ilişkisel modeli kullanan ve tablolar arasındaki ilişkileri tanımlayan veri tabanı türüdür. Her tablo, birincil anahtarlar ve yabancı anahtarlarla birbirine bağlanır. Örnek olarak, MySQL, Oracle, Microsoft SQL Server gibi veri tabanı yönetim sistemleri ilişkisel veri tabanı sağlar.

Nesne Tabanlı Veri tabanı: Nesne tabanlı veritabanları, nesne tabanlı programlamanın prensiplerini veri tabanına uygulayan bir türdür. Nesne tabanlı programlama dilleriyle uyumlu çalışır ve nesnelerin doğrudan saklanmasına olanak tanır.

Belge Veri tabanı: Belge veritabanları, dokümanlar ve koleksiyonlar şeklinde verileri saklamak için kullanılan bir türdür. Belge tabanlı veritabanları, JSON veya XML gibi belge formatlarını kullanarak verileri organize eder. MongoDB, CouchDB gibi veri tabanı yönetim sistemleri belge veri tabanı sağlar.

Graf Veri tabanı: Graf veritabanları, graf (graph) veri yapısını temel alan bir türdür. Graf veritabanları, düğümler (nodes) ve kenarlar (edges) arasındaki ilişkileri saklar. Bu tür veritabanları, ağlar, sosyal ağ analizi, coğrafi bilgi sistemleri gibi alanlarda kullanılır.

## 2.10 MSSQL Nedir ve Nerelerde Kullanılır?

MSSQL (Microsoft SQL Server), Microsoft tarafından geliştirilen ve ilişkisel veri tabanı yönetim sistemlerinden biridir. MSSQL, Windows işletim sistemlerinde çalışır ve genellikle büyük ölçekli işletmeler ve kuruluşlar tarafından kullanılır.

MSSQL, güvenilirlik, ölçeklenebilirlik ve performans gibi özellikleriyle öne çıkar. İlişkisel veri tabanı modelini kullanır ve birçok gelişmiş özelliği destekler. Örneğin, işlem yükünü dengeleme, yedekleme ve geri yükleme, yüksek kullanılabilirlik gibi özellikler sunar.

MSSQL, web uygulamaları, iş uygulamaları, e-ticaret siteleri, veri analizi projeleri, kurumsal yazılımlar gibi çeşitli alanlarda kullanılır. Özellikle büyük veri hacimlerinin saklanması, güncellenmesi ve sorgulanması gereken projelerde tercih edilen bir veri tabanıdır.

## 2.11 C# ve MSSQL Kullanarak Otomasyon Yapımı

C# programlama dili ve MSSQL veri tabanı, otomasyon projeleri geliştirmek için güçlü bir kombinasyon oluşturur. C#, nesne tabanlı programlama yetenekleri ve geniş kütüphane desteğiyle otomasyon projelerinin geliştirilmesini kolaylaştırır. MSSQL ise güvenilir ve ölçeklenebilir bir veri tabanı sağlayarak veri yönetimini ve iş süreçlerini destekler.

C# ve MSSQL kullanarak otomasyon projeleri, iş süreçlerini otomatikleştirmek, veri girişini ve analizini kolaylaştırmak, verimliliği artırmak için kullanılabilir. Örneğin, envanter yönetimi, müşteri ilişkileri yönetimi, personel takip sistemi gibi birçok alanda C# ve MSSQL tabanlı otomasyon projeleri geliştirilebilir.

C# dilinin zengin özellikleri ve Windows Form ile Visual Studio'nun sunduğu kolaylıklar, kullanıcı dostu arayüzlerin oluşturulmasını sağlar. MSSQL veri tabanı ise verilerin güvenli ve etkili bir şekilde yönetilmesini sağlar. Bu kombinasyon, otomasyon projelerinin geliştirilmesinde hız, güvenilirlik ve esneklik sağlar.

1. PROJE ANALİZİ: C# İLE ÖĞRENCİ OTOMASYONU GELİŞTİRME

## 3.1 Proje Gereksinimleri

Bu projenin odak noktası üniversitelerdir. Projenin temel amacı, üniversitelerin öğrenci, öğretmen ve memur süreçlerini otomatize etmektir. Giriş ekranında yer alan şifre ve kullanıcı adı doğrulama mekanizması sayesinde, kullanıcılar ilgili rollerine göre ayrıştırılır ve kendi arayüzlerine yönlendirilir. Bu sayede, kullanıcılar daha anlaşılabilir ve kullanılabilir bir arayüzden faydalanarak işlemlerini gerçekleştirebilirler.

Öğretmen, memur ve öğrenci arayüzleri, her bir kullanıcının kendi rollerine özgü işlemlere erişim sağlamaktadır. Öğretmenler, öğrenci notları girişi, ders seçimini onaylama ve öğrenci listeleme gibi işlemleri gerçekleştirebilirken, memurlar öğrenci kayıtlarıve öğretmen kayıtları gibi süreçleri yönetebilir. Öğrenciler ise ders seçimi, not takibi, dersleri görüntüleme gibi işlemleri kendi arayüzlerinden kolayca gerçekleştirebilirler.

Bu projenin temel gereksinimi, üniversite çalışanlarının ve öğrencilerin karmaşık süreçleri daha etkin ve verimli bir şekilde yönetebilmeleridir. Kullanıcıların daha anlaşılır ve kullanıcı dostu bir arayüz ile çalışmaları, işlemleri hızlı ve hatasız bir şekilde gerçekleştirmelerini sağlamaktadır. Bu sayede, üniversite personeli ve öğrenciler, zaman ve emek tasarrufu yaparak daha verimli bir çalışma ortamı oluşturabilmektedirler.

## 3.2 Proje Geliştirme Yaklaşımı ve Teknolojiler:

Bu projenin geliştirilmesi için nesne tabanlı bir programlama dili olan C# tercih edildi. C#, projenin gereksinimlerini karşılamak için geniş bir dil ve kütüphane desteği sunar. Ayrıca, .NET Framework üzerinde çalışan C# sayesinde, projenin Windows işletim sistemleriyle uyumlu bir şekilde çalışması sağlandı.

Windows Form, projede kullanıcı dostu ve basit bir arayüz sunmak için tercih edilen bir arayüz teknolojisidir. Windows Form, geliştiricilere hazır bileşenler ve kontroller sunarak arayüz tasarımını kolaylaştırır. Bu projede, kullanıcıların rahatlıkla etkileşimde bulunabileceği textbox, combobox, datagridview, label, imagelist, button gibi yapıları Windows Form üzerinde kullanıldı.

Projenin veri tabanı işlemleri için butonlarda kullanılan onclick fonksiyonlarıyla veri tabanına veri ekleme, güncelleme ve listeleme gibi işlemler gerçekleştirildi. Bu sayede, kullanıcılar ilgili butonlara tıkladıklarında arka planda çalışan fonksiyonlar sayesinde veri tabanı işlemlerini gerçekleştirebildiler. Veri tabanı işlemlerinin gerçekleştirilmesi için Microsoft SQL Server (MSSQL) tercih edildi. MSSQL, güvenilir ve büyük ölçekli veritabanları için etkili bir çözüm sunmaktadır.

## 3.3 Kullanılan C# Kütüphaneleri ve Projeye Eklenen Özellikler:

Proje geliştirme sürecinde, C# programlama dilinde çeşitli kütüphaneler kullanıldı. Bunlar, projenin işlevselliğini artırmak, veri tabanı işlemlerini yönetmek ve kullanıcı arayüzüyle etkileşimi sağlamak için önemli bir rol oynadı. Öncelikle, System kütüphanesi, temel C# türlerini ve işlevlerini içerir. Ayrıca, System.Data kütüphanesi, veri tabanı işlemleri için DataSet ve DataTable gibi yapıları sağlar.

Projede, System.Windows.Forms kütüphanesi kullanılarak Windows Form uygulaması oluşturuldu. Bu kütüphane, kullanıcı dostu ve interaktif bir arayüz sağlamak için çeşitli kontrolleri (textbox, combobox, datagridview, label, imagelist, button vb.) sunar. Ayrıca, System.Data.SqlClient kütüphanesi kullanılarak MSSQL veri tabanıyla iletişim kuruldu. Bu kütüphane, SQL sorguları oluşturmak, veri tabanına veri ekleme, güncelleme ve listeleme gibi işlemleri gerçekleştirmek için kullanıldı.

Proje içerisinde 10 adet form bulunmaktadır. Bunlar, öğrenci ekranı, öğretmen ekranı, memur ekranı, not görüntüleme ekranı, not ekleme ekranı, öğrenci-öğretmen güncelleme ve ekleme ekranları, ders seçimi ekranı, ders onay ekranı, şifre ve kullanıcı adı ile giriş ekranıdır. Bu formlar, farklı kullanıcı rollerine ve işlevlerine yönelik özelleştirilmiş arayüzleri sunar.

Veri tabanı tarafında 9 adet tablo kullanıldı ve bu tablolar arasında ilişkiler kuruldu. Öğrenci tablosu, öğretmen tablosu, memur tablosu, öğretmen-ders ilişkilerini tutan öğretmenders tablosu, öğrenci-ders ilişkilerini tutan öğrenciders tablosu, kullanıcı bilgilerini saklayan user tablosu, ders bilgilerini tutan ders tablosu, bölüm bilgilerini içeren bolum tablosu ve not bilgilerini saklayan not tablosu yer almaktadır.

Bu kütüphaneler ve eklenen özellikler sayesinde, projede kullanıcı dostu bir arayüz oluşturulmuş, veri tabanı işlemleri kolayca yönetilebilir hale getirilmiştir. Ayrıca, farklı kullanıcı rollerine özgü işlemlerin gerçekleştirildiği formlar ve ilişkili tablolar sayesinde proje daha işlevsel hale gelmiştir.

## 3.4 Proje Geliştirme Süreci ve Karşılaşılan Zorluklar

Projeyi geliştirme aşamasında, bazı zorluklarla karşılaşıldı. Özellikle combobox verilerini listeleme ve veri tabanı ilişkilendirme işlemleri, projenin geliştirilme sürecindeki karmaşık noktalardan biriydi. Combobox kontrolüne verileri doldurmak ve bu verilerle ilişkili işlemleri gerçekleştirmek, başlangıçta bazı zorluklar doğurdu. Ancak, araştırmalar yapılarak sorunlar aşıldı ve çözümler bulundu.

İlişkisel veri tabanı alanında ise, sütunların doğru şekilde ilişkilendirilmesi bazen karmaşık olabilirdi. Farklı tablolar arasındaki ilişkilerin kurulması ve bu ilişkilerin projenin gereksinimlerine uygun olarak çalışması için özenli bir çalışma gerektirdi. İlgili tablolardaki sütunların eşleştirilmesi ve uyumlu veri akışının sağlanması sürecinde, dikkatli bir şekilde ilerlendi ve hata kodlarından yola çıkılarak çözümler üretildi.

Projenin geliştirme sürecinde, fonksiyonların düzgün çalışması ve olası hataların tespit edilebilmesi için try-catch komutları kullanıldı. Bu şekilde, hata durumlarında programın çökmemesi ve kullanıcıya anlaşılır hata mesajları verilmesi sağlandı. Karşılaşılan problemler için detaylı araştırmalar yapıldı ve gerekli çözümler bulunarak projenin işlevselliği iyileştirildi.

Proje geliştirilirken kodlar tamamlandıktan sonra, tasarım alanında bazı değişiklikler yapıldı. Kullanıcıların daha anlaşılabilir ve kullanılabilir bir arayüz deneyimi yaşaması için görsel düzenlemeler gerçekleştirildi. Bu değişiklikler, projenin son halini daha estetik ve kullanıcı dostu bir şekilde sunmayı amaçladı.

Sonuç olarak, projenin geliştirme sürecinde bazı zorluklarla karşılaşıldı ancak bu zorluklar, araştırma ve çözüm odaklı yaklaşımlarla aşıldı. Kullanılan try-catch komutlarıyla hata yönetimi sağlandı ve projenin işlevselliği iyileştirildi. Ayrıca, tasarım alanında yapılan değişikliklerle kullanıcı deneyimi artırıldı.

4. DİZAYN

## 4.1 User Case Diagram

Formun Üstü

User Case Diagram 1

**metin, ekran görüntüsü, diyagram, daire içeren bir resim

Açıklama otomatik olarak oluşturuldu**

Öğrenci Ekranı:

1. Kişisel Bilgileri Görüntüleme: Öğrencinin kişisel bilgilerini görüntüleme yeteneği.
2. Ders Bilgilerini Görüntüleme: Öğrencinin derslere ilişkin bilgileri görüntüleme yeteneği.
3. Ders Seçim Ekranına Git: Öğrencinin ders seçim ekranına erişim yeteneği.
4. Not Görüntüleme: Öğrencinin notlarını görüntüleme yeteneği.
5. Dersleri Listeleme: Öğrencinin dersleri listeleme yeteneği.
6. Ders Ekleme: Öğrencinin yeni bir ders ekleyebilme yeteneği.

Öğretmen Ekranı:

1. Kişisel Bilgileri Görüntüleme: Öğretmenin kişisel bilgilerini görüntüleme yeteneği.
2. Danışman Onayı: Öğretmenin danışmanlık onayı verme yeteneği.
3. Not Giriş Ekranı: Öğretmenin not girişi yapabilme yeteneği.
4. Danışmanlık Yapılan Öğrenciler: Öğretmenin danışmanlık yaptığı öğrencileri görüntüleme yeteneği.
5. Dersleri Listeleme: Öğretmenin dersleri listeleme yeteneği.

Memur Ekranı:

1. Kişisel Bilgileri Görüntüleme: Memurun kişisel bilgilerini görüntüleme yeteneği.
2. Kişi Güncelleme: Memurun kişi bilgilerini güncelleme yeteneği.
3. Kişi Ekleme: Memurun yeni bir kişi ekleyebilme yeteneği.

## 4.2 Veri tabanı er diyagramları şema tasarımları ve tablolar

Sistemde 3 tip kullanıcı bulunmaktadır: "Öğrenci", "Öğretmen" ve "Memur".

Öğrenciler, kendi özelliklerine ek olarak "Bölüm" ve "Öğretmen" tablolarıyla ilişkili oldukları için her iki tablonun anahtar özelliklerini kendi tablolarına eklemektedirler.

Bir "Öğrenci" bir veya birden fazla ders alabileceği için ve bir "Ders" birden fazla öğrenci tarafından alınabileceği için "Çoktan Çok'a" ilişkisi içerisindedirler. Bu nedenle her iki tablonun anahtar özellikleri birleştirilerek, ilişki özelliklerini de dahil ederek yeni bir "ÖğrenciDersTablosu" oluşturulmaktadır.

Her "Öğretmen" bir veya birden fazla dersi verebileceği için ve bir "Ders" bir veya birden fazla "Öğretmen" tarafından verilebileceği için her iki tablonun anahtar özellikleri yeni bir tablo oluşturmamaktadır.

Her "Öğrenci" ve "Öğretmen" bir "Bölüm"e bağlı olmak zorunda oldukları için "Bölüm" tablosundan yabancı anahtar almaktadırlar.

metin, diyagram, ekran görüntüsü, paralel içeren bir resim

Açıklama otomatik olarak oluşturuldu

Veri tabanı İlişkisel Diagram 1

"Öğrenciler" aldıkları her "Ders" (ÖğrenciDersTablosu) için bir notlandırma kriterine sahip olduğu için "ÖğrenciDersTablosu" "NotTablosu"na yabancı anahtar vermektedir."Memur" ve "KullanıcıTablosu" tabloları herhangi bir ilişki içinde değillerdir.

## Tablolar

## BolumTable

Tablo 1

**metin, ekran görüntüsü, sayı, numara, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu**

BolumTable adlı tabloda iki sütun bulunmaktadır. BolumID sütunu tablonun Birincil anahtarıdır ve BolumAd sütunundaki bölümleri maksimum 3 karakter kullanılacak şekilde kısaltır. BolumID sütunu Bölüm kodu işlevi görmektedir.

## DersTable

Tablo 2

**metin, ekran görüntüsü, sayı, numara, menü içeren bir resim

Açıklama otomatik olarak oluşturuldu**

DersTable tablosunda toplam dört sütun bulunmaktadır. DersID sütunu tablonun birincil anahtarıdır. Bunun yanında DersAdi, DersKredi ve DersAKTS sütunlarıda mevcuttur. DersAdi adından da anlaşılacağı üzere Ders isimlerini barındıran sütundur. DersKredi sütunu derslerin kaç krediye sahip olduğunu belirtir. DersAKTS ise derslerin AKTS bazında kaç kredi olduğunu temsil eder.

## MemurTable

**metin, ekran görüntüsü, yazı tipi, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturuldu**

Tablo 3

Memur ( Admin ) bilgilerinin tutulduğu tablodur. MemurTable tablosu toplamda beş sütundan oluşmaktadır. MemurID tablonun birincil anahtarıdır ve 39060299 sayısından büyük değer alacak şekilde ayarlandı.

MemurAd, MemurSoyad, MemurTC, MemurSifre sütunları, sırasıyla Memur’un adını, soyadını, Türkiye Cumhuriyeti kimlik numarasını ve Şifre bilgilerini saklı tutar.

## NotTable

metin, ekran görüntüsü, sayı, numara, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

Tablo 4

Öğrenciler için not bilgilerinin tutulduğu tablodur. Toplamda sekiz sütundan oluşmaktadır. NotID tablonun birincil anahtarıdır. OgrenciDersID, OgrenciDersTable tablosundaki birincil anahtar olan OgrenciDersID sütunu ile ilişkilendirildi. Vize, Final, Butunleme sütunları Öğretmen tarafndan girilen notları hesaplar ve not durumuna göre GecmeNotu ve GecmeDurumu sütunları doldurulur. Notlar farklı zamanlarda girilebileceği için not sütunları null değeri alabilir. OgrenciDonem sütunu ise öğrencinin okulda bulunduğu dönem sayısını ifade eder.

## metin, sayı, numara, yazı tipi, çizgi içeren bir resim Açıklama otomatik olarak oluşturulduOgrenciTable

Tablo 5

Öğrenci bilgilerinin saklandığı tablodur. Toplam sekiz sütundan oluşmaktadır. Tablo’nun birincil anahtarı OgrenciNo sütunudur. OgrenciAd, OgrenciSoyad, OgrenciTC sırasıyla öğrenci’nin adını,soyadını ve Türkiye Cumhuriyeti kimlik numarasını saklı tutar. OgrenciBolumID, BolumTable tablosundaki BolumID ile ilişkilendirildi. OgrenciDonem NotTable tablosundaki OgrenciDonem sütunu ile ilişkilendirildi ve Ogrencinin dönem sayısını saklar. OgrenciDanismanID sütunu ise OgretmenDersTable tablosunda bulunan OgretmenID ile ilişkilendirildi ve öğretmen numarasını temsil eder. OgrenciSifre sütunu ise öğrencinin şifre bilgilerini saklı tutar.

## OgretmenDersTable

metin, ekran görüntüsü, yazı tipi, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturuldu

Tablo 6

Öğretmen ve derslerin ilişkilendirilmesini sağlayan tablodur. Toplam üç sütundan oluşmaktadır. Birincil anahtarı OgretmenDersID sütunudur. OgretmenID sütunu öğretmen numaralarını saklamaktadır. DersID sütunu ise DersTable tablosunda bulunan DersID sütunu ile ilişkilendirilmiştir.

## metin, sayı, numara, yazı tipi, yazılım içeren bir resim Açıklama otomatik olarak oluşturulduOgretmenTable

Tablo 7

Öğretmen bilgilerinin tutulduğu tablodur. Toplam altı sütundan oluşmaktadır. Birincil anahtarı öğretmen numarasını temsil eden OgretmenID sütunudur. OgretmenAd, OgetmenSoyad sütunları öğretmenin adını ve soyadını saklı tutar. OgtretmenBolumID sütunu ise BolumTable tablosundaki BolumID sütunundan seçilir. OgretmenTC sütunu öğretmenin Türkiye Cumhuriyeti kimlik bilgilerini saklı tutar. OgretmenSifre sütunu ise Öğretmenin şifre bilgilerini saklar.

Tablo 8

metin, ekran görüntüsü, sayı, numara, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

## 4.3.8 UserTable

Tablo tek sütundan oluşur ve Kullanıcıların Türkiye Cumhuriyeti kimlik numaralarını saklar. Aynı kimlik numarasını ikinci kez kullanımını önler.

## OgrenciDersTable

Tablo 9

metin, ekran görüntüsü, sayı, numara, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

Tablo beş sütundan oluşmaktadır. Tablo OgrenciTable ve DersTable tablolarının birbiri ile ilişkilendirilmesini ve ders kayıt işlemlerinin yürütülmesini sağlamaktadır. Birincil anahtarı OgrenciDersID sütunudur. Her öğrenciye ait olan ders farklı bir OgrenciDersID sütunuyla tutulmaktadır. OgrenciNo, DersID, OgrenciDersDonem, DanismanOnay sütunları sırasıyla öğrenci numarasını, öğrencilerin derslerini, öğrenci dönem dersini ve danışman onay durumunu saklamaktadır.

**5. TASARIM**

## 5.1 Giriş Paneli

Form 1

metin, ekran görüntüsü, logo, grafik içeren bir resim

Açıklama otomatik olarak oluşturuldu

Giriş sayfası kullanıcıdan alınan şifre ve öğretmen, öğrenci veya admin numarası girildikten sonra kullanıcıya özel olan sayfaya yönlendirilir. Basit ve kullanışlı bir arayüz için bir groupbox, bir button, iki textbox, bir imagelist ve bir adet picturebox kullanıldı.

## 5.2 Memur Paneli

Form 2

metin, ekran görüntüsü, yazı tipi, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturulduMemu Panelinde bir groupbox, üç textbox, 3 label ve iki button kullanıldı. Anlaşılır bir arayüz için butonlara ikonlar eklendi. Memur tetboxlarında değişiklik yapılamayacak şekilde memur adı, soyadı ve numarası gösterildi. Kişi ekleme butonu öğrenci ve öğretmen ekleme için kullanılan sayfaya yönlendirildi. Kişi Güncelleme butonu ise Öğretmen ve öğrenci bilgilerinin güncellendiği sayfaya yönlendirildi.

## metin, ekran görüntüsü, yazı tipi, sayı, numara içeren bir resim Açıklama otomatik olarak oluşturuldu5.3 Kişi Ekleme Paneli

Form 3

Öğrenci Ekleme tablosunda iki groupbox, 3 combobox, sekiz textbox, iki buton ve onbir adet label kullanıldı. İş yükünü azaltmak adına öğretmen ve öğrenci bilgileri ekleme işlemi tek bir formda yapıldı.Öğrenci Ekle butonu OgrenciTable tablosuna, Öğretmen Ekle butonu ise OgretmenTable tablosuna verileri ekler. Öğrenci Bölümü ve Öğretmen Branşı comboboxlarında üç harften oluşan ders kodları yer alır. Danışman numarası comboboxı ise öğretmen numaralarını listeler.

## metin, ekran görüntüsü, yazılım, web sayfası içeren bir resim Açıklama otomatik olarak oluşturuldu5.4 Kişi Güncelleme Paneli

Form 4

Kişi Güncelle Panelinde iki groupbox, beş button, iki datagridview, ondört textbox ve ondört label kullanıldı.Datagridviewda onClick metotu kullanılarak bir hücreye tıklandığında boş olan textboxlar tıklanılan satırdaki kullanıcının bilgileri ile dolacak şekilde ayarlandı. Öğrenci Listele ve Öğretmen Listele butonları OgrenciTable ve OgretmenTable tablolarında bulunan verileri datagridviewda listeler. Öğrenci Güncelle ve Öğretmen Güncelle butonları ise sırasıyla OgrenciTable ve OgretmenTable tablolarında bulunan verileri günceller. Öğrenci No ve Öğretmen No verileri Birincil anahtar oldukları için güncellenemez.Dönem bitir butonu ise dönemi sonlandırmak için kullanılır.

## 5.5 Öğretmen Paneli

Form 5

metin, ekran görüntüsü, sayı, numara, yazılım içeren bir resim

Açıklama otomatik olarak oluşturuldu

Öğretmen Panelinde üç groupbox, dört label, dört textbox, iki datagridview, iki button kullanıldı. Derslerim groupboxunda bulunan datagridviewda öğretmenin sorumlu olduğu dersler listelendi. Diğer datagridviewda ise Danışmanlık yapılan öğrenciler listelendi. Danışman onay butonuna tıklandıktan sonra ders seçimi yapan öğrencilerin derslerini onaylamak için ders onayı ekranı açılır. Not girişi butonunda ise öğrencilerin vize, final ve bütünleme notlarının girileceği not ekleme sayfası açılır.

## 5.6 Ders Onay Paneli

metin, ekran görüntüsü, yazılım, işletim sistemi içeren bir resim

Açıklama otomatik olarak oluşturuldu

Form 6

Öğretmen Panelinden erişim sağlanır. Ders onayı sayfasında iki adet buton, bir adet datagridview ve bir adet groupbox kullanıldı.Bu sayfada öğrencilerin almak istediği dersler seçildikten sonra öğretmen onayını alacağı sayfadır. Kaydı onayla butonu öğrencinin seçtiği dersi onaylarken kaydı reddet butonu, öğrencinin dersi almasını reddedecektir. Öğrencilerin seçim yaptığı dersler bu tabloda listelenir.

## 5.7 Not Ekleme Paneli

metin, ekran görüntüsü, yazılım, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturulduForm 7

Öğretmen Panelinden erişilen Not ekleme sayfası toplamda iki groupbox, üç textbox, üç label, bir button ve bir datagridviewdan oluşmaktadır. Öğretmenden öğrencinin Vize, Final ve Bütünleme not bilgilerini alarak öğrencinin geçme notunu ve geçme durumunu datagridviewa yansıtır. Girilmek istenen not girildikten sonra hesaplamak için Sisteme gir butonuna tıklanmalıdır.

## metin, ekran görüntüsü, sayı, numara, yazılım içeren bir resim Açıklama otomatik olarak oluşturuldu5.8 Öğrenci Paneli

Form 8

Giriş panelinden öğrenci paneline yönlendirilmek için kullanıcının öğrenci numarası ve şifresini girmesi gerekmektedir. Öğrenci Paneli toplamda üç groupbox, bir datagridview, sekiz label, sekiz textbox ve üç buttondan oluşmaktadır. Datagridviewda onClick metotu kullanılarak seçilen dersleri Ders Bilgileri groupboxunda bulunan textboxlara yazdırır. Panelde bulunan textboxlar sadece veriyi gösterir. Kullanıcı bu verileri değiştiremez. Datagridviewda ders seçildikten sonra Ders ekle butonuna tıklanır. Daha sonra Ders seçim ekranına gir butonuna tıklanır ve seçilen dersler bu sayfada danışman onayına gönderilir. Not görüntüle butonu ise öğrencinin öğretmenler tarafından girilen notlarını listeler.

## 5.9 Ders Seçim Paneli

metin, ekran görüntüsü, sayı, numara, yazılım içeren bir resim

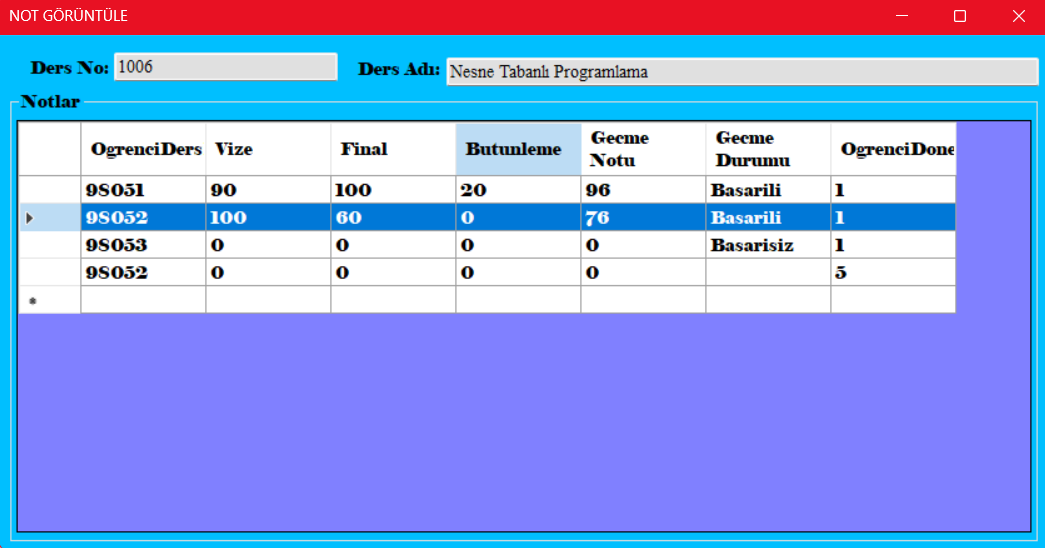
Açıklama otomatik olarak oluşturuldu

Form 9

Öğrenci sayfasında bulunan derslerden seçim yapılıp Ders ekle butonuna tıklandıktan sonra veriler bu sayfada listelenir. Ders seçimi panelinde toplamda bir groupbox, bir datagridview ve iki adet button bulunmaktadır. Datagridviewdan seçim yapabilmek için onClick metotu eklenmiştir. Ders seçildikten sonra Danışmana gönder butonuna tıklandığı takdirde Ders sadece danışman öğretmenin erişim sağlayacağı ders onay paneline gönderilir ve danışman burada öğrencinin dersi alıp alamayacağına karar verebilir. Listeden kaldır butonu kullanıldığında ise seçilen ders listeden kaldırılır.

## 5.10 Not Görüntüleme Paneli

Form 10



Öğrenci Panelinde bulunan not görüntüle butonuna tıklandıktan sonra bu sayfaya yönlendirme yapılır. Not Görüntüleme paneli toplamda iki label, iki textbox, bir groupbox ve bir adet datagridviewdan oluşmaktadır. Datagridviewda onClick metotu kullanılarak seçim yapıldıktan sonra ders adı ve ders numarası textboxlara yazdırılır. Textboxlar sadece okumaya açıktır. Üzerinde değişiklik yapılamaz. Bu sayfada sadece öğrenciye göre NotTable tablosundan veri listelenir.

1. KODLAMA

## 6.1 Kullanılan kütüphaneler

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Data.SqlClient;

using System;: Bu kütüphane, C# programlama dilinin temel sistem türlerini ve işlevlerini içerir. Bu türler ve işlevler, genel amaçlı veri türleri, matematiksel işlemler, hata işleme, dizeler, giriş/çıkış işlemleri ve diğer sistem düzeyi işlemleri gibi temel programlama işlevlerini sağlar.

using System.Collections.Generic;: Bu kütüphane, C#'da genel koleksiyon türleri ve işlevlerini içerir. Bu kütüphane, listeler, sözlükler, kuyruklar, yığınlar ve diğer koleksiyonlar gibi veri yapılarını ve bu yapılar üzerindeki işlemleri sağlar.

using System.ComponentModel;: Bu kütüphane, bileşen tabanlı programlama için gerekli olan bileşen modeli türlerini içerir. Bu türler, bileşenlerin yaşam döngüsü, olaylar, bağımlılıklar ve diğer bileşen tabanlı işlevleri yönetmek için kullanılır.

using System.Data;: Bu kütüphane, veri tabanı işlemleri için gerekli olan temel ADO.NET türlerini içerir. Bu kütüphane, veri tabloları, veri kümeleri, veri görünümleri ve diğer veri yapılarıyla çalışmak için kullanılır.

using System.Drawing;: Bu kütüphane, grafikler, çizimler ve kullanıcı arayüzü bileşenleri için gerekli olan grafik işlemleri ve türleri içerir. Bu kütüphane, çizimler yapma, renklerle çalışma, resim işleme ve kullanıcı arayüzü tasarlama gibi grafik tabanlı işlevleri sağlar.

using System.Linq;: Bu kütüphane, LINQ (Language Integrated Query) sorguları için gerekli olan türleri ve işlevleri içerir. Bu kütüphane, veri koleksiyonları üzerinde sorgulama, filtreleme, sıralama ve dönüşüm işlemleri yapmak için kullanılır.

using System.Text;: Bu kütüphane, metin işleme ve kodlamalar için gerekli olan türleri ve işlevleri içerir. Bu kütüphane, dize manipülasyonu, karakter kodlamaları, metin biçimlendirme ve diğer metin tabanlı işlevleri sağlar.

using System.Threading.Tasks;: Bu kütüphane, çoklu iş parçacığı ve eşzamanlılık işlevleri için gerekli olan türleri ve işlevleri içerir. Bu kütüphane, paralel programlama, asenkron işlemler ve diğer çoklu iş parçacığı senaryoları için kullanılır.

using System.Windows.Forms;: Bu kütüphane, Windows Forms uygulamaları için gerekli olan kullanıcı arayüzü bileşenlerini içerir. Bu kütüphane, form oluşturma, düğmeler, metin kutuları, liste kutuları ve diğer kullanıcı arayüzü bileşenlerinin oluşturulması ve etkileşimi için kullanılır.

using System.Data.SqlClient;: Bu kütüphane, Microsoft SQL Server veri tabanıyla etkileşimde bulunmak için gerekli olan türleri ve işlevleri içerir. Bu kütüphane, SQL Server veri tabanına bağlanma, sorgu gönderme, veri alma ve güncelleme işlemlerini sağlar.

## 6.2 girisPanel.cs

private SqlConnection baglanti = new SqlConnection("Data Source=(localdb)\\ProjectModels;Initial Catalog=OgrenciOtomasyon;Integrated Security=True");

SqlConnection sınıfından baglanti adında özel bir nesne oluşturulur ve belirtilen veri tabanına bağlanmak için gerekli bağlantı dizesi kullanılır.

private void buttonGiris\_Click(object sender, EventArgs e)

{

form\_variable = txbKullaniciAdi.Text;

if (baglanti.State.ToString() == "Closed")

{

baglanti.Open();

}

try

{

if (int.Parse(txbKullaniciAdi.Text.ToString()) > 19060299 && int.Parse(txbKullaniciAdi.Text.ToString()) < 29060300)

{

string query = "SELECT \* FROM OgrenciTable WHERE OgrenciNo = @No AND OgrenciSifre = @Sifre";

SqlParameter prm1 = new SqlParameter("@No", txbKullaniciAdi.Text.Trim());

SqlParameter prm2 = new SqlParameter("@Sifre", txbSifre.Text.Trim());

SqlCommand cmd = new SqlCommand(query, baglanti);

cmd.Parameters.Add(prm1);

cmd.Parameters.Add(prm2);

DataTable dataTable = new DataTable();

SqlDataAdapter adapter = new SqlDataAdapter(cmd);

adapter.Fill(dataTable);

if (dataTable.Rows.Count > 0)

{

Form f = new ogrenciPanel();

f.Show();

}

else

{

MessageBox.Show("Kullanıcı bulunamadı");

}

}

else if (int.Parse(txbKullaniciAdi.Text.ToString()) > 29060299 && int.Parse(txbKullaniciAdi.Text.ToString()) < 39060300)

{

string query = "SELECT \* FROM OgretmenTable WHERE OgretmenID = @ID AND OgretmenSifre = @Sifre";

SqlParameter prm1 = new SqlParameter("@ID", txbKullaniciAdi.Text.Trim());

SqlParameter prm2 = new SqlParameter("@Sifre", txbSifre.Text.Trim());

SqlCommand cmd = new SqlCommand(query, baglanti);

cmd.Parameters.Add(prm1);

cmd.Parameters.Add(prm2);

DataTable dataTable = new DataTable();

SqlDataAdapter adapter = new SqlDataAdapter(cmd);

adapter.Fill(dataTable);

if (dataTable.Rows.Count > 0)

{

Form f = new ogretmenPanel();

f.Show();

}

else

{

MessageBox.Show("Kullanıcı bulunamadı");

}

}

else if (int.Parse(txbKullaniciAdi.Text.ToString()) > 39060299)

{

string query = "SELECT \* FROM MemurTable WHERE MemurID = @ID AND MemurSifre = @Sifre";

SqlParameter prm1 = new SqlParameter("@ID", txbKullaniciAdi.Text.Trim());

SqlParameter prm2 = new SqlParameter("@Sifre", txbSifre.Text.Trim());

SqlCommand cmd = new SqlCommand(query, baglanti);

cmd.Parameters.Add(prm1);

cmd.Parameters.Add(prm2);

DataTable dataTable = new DataTable();

SqlDataAdapter adapter = new SqlDataAdapter(cmd);

adapter.Fill(dataTable);

if (dataTable.Rows.Count > 0)

{

Form f = new memurPanel();

f.Show();

}

else

{

MessageBox.Show("Kullanıcı bulunamadı");

}

}

else

{

if (txbKullaniciAdi.Text.Length != 11 || txbSifre.Text.Length != 8)

{

MessageBox.Show("Geçersiz Kullanıcı Adı veya Şifre");

}

}

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "");

}

}

buttonGiris\_Click adlı metot, "Giriş" düğmesine tıklandığında çalışan olay işleyicisidir. Aşağıdaki adımları içerir:

form\_variable değişkeni, txbKullaniciAdi metin kutusunun içeriğiyle eşitlenir.

Bağlantı durumu "Closed" ise, bağlantı açılır.

try bloğu içinde veri tabanından kullanıcı bilgilerini sorgulamak için if-else yapısı kullanılır:

a. Eğer txbKullaniciAdi değeri 19060299 ile 29060300 arasında ise, "SELECT \* FROM OgrenciTable WHERE OgrenciNo = @No AND OgrenciSifre = @Sifre" sorgusu oluşturulur.

b. Eğer txbKullaniciAdi değeri 29060299 ile 39060300 arasında ise, "SELECT \* FROM OgretmenTable WHERE OgretmenID = @ID AND OgretmenSifre = @Sifre" sorgusu oluşturulur.

c. Eğer txbKullaniciAdi değeri 39060299'dan büyükse, "SELECT \* FROM MemurTable WHERE MemurID = @ID AND MemurSifre = @Sifre" sorgusu oluşturulur.

Oluşturulan sorgu parametreleri ve bağlantı kullanılarak veri tabanından veri çekilir ve bir DataTable nesnesiyle doldurulur.

Eğer dataTable.Rows.Count değeri 0'dan büyükse, ilgili kullanıcı paneli açılır (ogrenciPanel, ogretmenPanel, memurPanel).

Eğer dataTable.Rows.Count değeri 0 ise, "Kullanıcı bulunamadı" mesajı gösterilir.

Eğer txbKullaniciAdi veya txbSifre metin kutularının uzunlukları 11 veya 8 değilse, "Geçersiz Kullanıcı Adı veya Şifre" mesajı gösterilir.

catch bloğu içinde herhangi bir hata durumunda hata mesajı gösterilir.

## 6.3 memurPanel.cs

private SqlConnection baglanti = new SqlConnection("Data Source=(localdb)\\ProjectModels;Initial Catalog=OgrenciOtomasyon;Integrated Security=True");

SqlConnection sınıfından baglanti adında özel bir nesne oluşturulur ve belirtilen veri tabanına bağlanmak için gerekli bağlantı dizesi kullanılır.

private void Form5\_Load(object sender, EventArgs e)

{

txbNo.Text = girisPanel.form\_variable;

baglanti.Open();

string query = "select \* from MemurTable where MemurID = '" + txbNo.Text + "'";

SqlCommand cmd = new SqlCommand(query, baglanti);

SqlDataReader reader = cmd.ExecuteReader();

reader.Read();

txbAdi.Text = reader.GetString(1);

txbSoyadi.Text = reader.GetString(2);

reader.Close();

baglanti.Close();

}

"txbNo" adlı TextBox bileşenine, "girisPanel" formundan aktarılan "form\_variable" değeri atanır. Bu değer, giriş yapan memurun ID'sini temsil etmektedir.

"baglanti" adlı SqlConnection nesnesi oluşturularak veri tabanına bağlantı açılır. Bağlantı dizesi, yerel veri tabanı sunucusuna (localdb)\ProjectModels üzerinden bağlanmayı sağlar.

"query" adlı bir sorgu dizesi oluşturulur. Bu sorgu, "MemurTable" tablosundan "MemurID" değeri "txbNo" TextBox'ının değerine eşit olan memurun verilerini seçer.

SqlCommand nesnesi olan "cmd" oluşturulur ve sorgu dizesi ile SqlConnection nesnesi bağlantısı ilişkilendirilir.

SqlDataReader nesnesi olan "reader" oluşturulur ve SqlCommand nesnesi ile veriler okunur.

"reader.Read()" metodu çağrılarak okunan veri satırına geçilir.

Okunan veriler, sırasıyla "txbAdi" ve "txbSoyadi" adlı TextBox bileşenlerine atanır. Okunan verilerin sırası, "reader.GetString(1)" ile 1. sütunu (Ad) ve "reader.GetString(2)" ile 2. sütunu (Soyad) temsil eder.

"reader.Close()" metodu ile SqlDataReader nesnesi kapatılır.

"baglanti.Close()" metodu ile SqlConnection nesnesi kapatılır ve veri tabanı bağlantısı sonlandırılır.

Bu işleyici metot, "memurPanel" formu yüklendiğinde belirtilen memurun verilerini veri tabanından okuyarak, "txbAdi" ve "txbSoyadi" adlı TextBox bileşenlerine yerleştirir. Böylece, memurun adı ve soyadı formda görüntülenir.

private void button1\_Click(object sender, EventArgs e)

{

Form f = new kisiEklePanel();

f.Show();

}

private void button2\_Click(object sender, EventArgs e)

{

Form f = new kisiGuncellePanel();

f.Show();

}

"button1\_Click" metodu:

Bir "kisiEklePanel" formu olan "f" değişkeni oluşturulur.

Oluşturulan form, "f.Show()" metodunu kullanarak görüntülenir. Bu şekilde, "kisiEklePanel" formu kullanıcıya gösterilir.

"button2\_Click" metodu:

Bir "kisiGuncellePanel" formu olan "f" değişkeni oluşturulur.

Oluşturulan form, "f.Show()" metodunu kullanarak görüntülenir. Bu şekilde, "kisiGuncellePanel" formu kullanıcıya gösterilir.

Yukarıdaki kod parçası, "button1" düğmesi tıklandığında "kisiEklePanel" formunu açar ve "button2" düğmesi tıklandığında ise "kisiGuncellePanel" formunu açar. Bu sayede kullanıcı, ilgili formlara erişerek belirli işlemleri gerçekleştirebilir.

## 6.4 kisiEklePanel.cs

private SqlConnection baglanti = new SqlConnection("Data Source=(localdb)\\ProjectModels;Initial Catalog=OgrenciOtomasyon;Integrated Security=True");

SqlConnection sınıfından baglanti adında özel bir nesne oluşturulur ve belirtilen veri tabanına bağlanmak için gerekli bağlantı dizesi kullanılır.

void BolumIDgetir()

{

baglanti.Open();

string query = "select BolumID from BolumTable";

SqlCommand cmd = new SqlCommand(query, baglanti);

SqlDataReader reader = cmd.ExecuteReader();

reader.Read();

while (reader.Read())

{

cxbOgrenciBolumu.Items.Add(reader.GetString(0));

cxbOgretmenBransi.Items.Add(reader.GetString(0));

}

reader.Close();

baglanti.Close();

}

Bağlantıyı aç: "baglanti.Open()" metodu ile veri tabanı bağlantısı açılır.

Sorguyu tanımla: "select BolumID from BolumTable" sorgusu "query" adlı değişkene atanır. Bu sorgu, "BolumTable" tablosundaki "BolumID" sütununu seçer.

Komut nesnesini oluştur: "SqlCommand" sınıfından "cmd" adında bir komut nesnesi oluşturulur. Bu nesne, sorguyu ve bağlantıyı kullanarak veri tabanı işlemlerini gerçekleştirir.

Verileri oku: "cmd.ExecuteReader()" metoduyla sorgu çalıştırılır ve sonuçlar "reader" adlı SqlDataReader nesnesine aktarılır.

Verileri oku ve ComboBox'lara ekle: "reader.Read()" metoduyla okuma işlemi gerçekleştirilir. Daha sonra, döngü kullanarak her bir kayıt için "cxbOgrenciBolumu" ve "cxbOgretmenBransi" ComboBox nesnelerine "BolumID" değerleri eklenir.

Veri tabanı kaynaklarını kapat: Okuma işlemi tamamlandıktan sonra "reader.Close()" ve bağlantıyı kapatmak için "baglanti.Close()" metodları kullanılır. Bu, veri tabanı kaynaklarının doğru bir şekilde serbest bırakılmasını sağlar.

Bu kod parçası, "BolumTable" tablosundan alınan "BolumID" değerlerini "cxbOgrenciBolumu" ve "cxbOgretmenBransi" ComboBox nesnelerine ekleyerek kullanıcıya seçenekler sunar.

private void button1\_Click(object sender, EventArgs e)

{

try

{

baglanti.Open();

SqlCommand sqlCommand2 = new SqlCommand("Insert into USERTable (USERTC) values ('" + txbOgrenciTC.Text + "')", baglanti);

sqlCommand2.ExecuteNonQuery();

baglanti.Close();

baglanti.Open();

string query = "Insert into OgrenciTable (OgrenciAd,OgrenciSoyad,OgrenciTC,OgrenciBolumID,OgrenciDanismanID,OgrenciSifre) values('" + tbxOgrenciAdi.Text + "','" + txbOgrenciSoyadi.Text + "','" + txbOgrenciTC.Text + "','" + cxbOgrenciBolumu.Text + "','" + Convert.ToInt32(cxbDanismanNo.Text) + "','" + txbOgrenciSifre.Text + "')";

SqlCommand sqlCommand = new SqlCommand(query, baglanti);

sqlCommand.ExecuteNonQuery();

baglanti.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

finally

{

baglanti.Close();

}

MessageBox.Show("Öğrenci Eklendi!");

cxbDanismanNo.Items.Clear();

cxbOgretmenBransi.Items.Clear();

cxbOgrenciBolumu.Items.Clear();

tbxOgrenciAdi.Clear();

txbOgrenciSoyadi.Clear();

txbOgrenciTC.Clear();

txbOgrenciSifre.Clear();

BolumIDgetir();

}

Bağlantıyı aç: "baglanti.Open()" metodu ile veri tabanı bağlantısı açılır.

İlk sorguyu çalıştır: "SqlCommand" sınıfından "sqlCommand2" adında bir komut nesnesi oluşturulur. Bu nesne, "USERTable" tablosuna "USERTC" sütununa "txbOgrenciTC.Text" değerini ekleyen bir INSERT sorgusunu yürütür. "ExecuteNonQuery()" metodu, sorguyu veri tabanında çalıştırır.

İkinci sorguyu çalıştır: "string" türünde "query" adında bir değişken tanımlanır ve "OgrenciTable" tablosuna öğrenci bilgilerini eklemek için bir INSERT sorgusu oluşturulur. Değişkenler, ilgili TextBox ve ComboBox nesnelerinin değerlerini kullanarak doldurulur. Ardından, "SqlCommand" sınıfından "sqlCommand" adında bir komut nesnesi oluşturulur ve sorgu "sqlCommand.ExecuteNonQuery()" metoduyla veri tabanında çalıştırılır.

Veri tabanı kaynaklarını kapat: Sorgular başarıyla çalıştırıldıktan sonra "baglanti.Close()" metoduyla veri tabanı bağlantısı kapatılır. Bu, veri tabanı kaynaklarının serbest bırakılmasını sağlar.

Bilgi mesajı göster: "MessageBox.Show()" metoduyla kullanıcıya "Öğrenci Eklendi!" mesajı gösterilir.

ComboBox ve TextBox nesnelerini temizle: Öğrenci ekleme işleminden sonra ilgili ComboBox ve TextBox nesneleri temizlenir.

Bölüm ID'leri yeniden yükle: "BolumIDgetir()" metodu çağrılarak ComboBox nesnelerine güncel bölüm ID'leri eklenir.

Bu kod parçası, öğrenci bilgilerini alarak veri tabanına kaydetmeyi ve ardından kullanıcıya geri bildirim vermek ve ComboBox ve TextBox nesnelerini temizlemek için gerekli işlemleri gerçekleştirir.

private void button2\_Click(object sender, EventArgs e)

{

try

{

baglanti.Open();

SqlCommand sqlCommand2 = new SqlCommand("Insert into USERTable (USERTC) values ('" + txbOgretmenTC.Text + "')", baglanti); // Aynı şekilde öğretmen ekleme

sqlCommand2.ExecuteNonQuery();

baglanti.Close();

baglanti.Open();

string query = "Insert into OgretmenTable (OgretmenAd,OgretmenSoyad,OgretmenBolumID,OgretmenTC,OgretmenSifre) values('" + txbOgretmenAdi.Text + "','" + txbOgretmenSoyadi.Text + "','" + cxbOgretmenBransi.Text + "','" + txbOgretmenTC.Text + "','" + txbOgretmenSifre.Text + "')";

SqlCommand sqlCommand = new SqlCommand(query, baglanti);

sqlCommand.ExecuteNonQuery();

baglanti.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

baglanti.Close();

}

MessageBox.Show("Öğretmen Eklendi!");

txbOgretmenAdi.Clear();

txbOgretmenSoyadi.Clear();

txbOgretmenTC.Clear();

cxbOgretmenBransi.Items.Clear();

txbOgretmenSifre.Clear();

}

Bağlantıyı aç: "baglanti.Open()" metodu ile veri tabanı bağlantısı açılır.

İlk sorguyu çalıştır: "SqlCommand" sınıfından "sqlCommand2" adında bir komut nesnesi oluşturulur. Bu nesne, "USERTable" tablosuna "USERTC" sütununa "txbOgretmenTC.Text" değerini ekleyen bir INSERT sorgusunu yürütür. "ExecuteNonQuery()" metodu, sorguyu veri tabanında çalıştırır.

İkinci sorguyu çalıştır: "string" türünde "query" adında bir değişken tanımlanır ve "OgretmenTable" tablosuna öğretmen bilgilerini eklemek için bir INSERT sorgusu oluşturulur. Değişkenler, ilgili TextBox ve ComboBox nesnelerinin değerlerini kullanarak doldurulur. Ardından, "SqlCommand" sınıfından "sqlCommand" adında bir komut nesnesi oluşturulur ve sorgu "sqlCommand.ExecuteNonQuery()" metoduyla veri tabanında çalıştırılır.

Veri tabanı kaynaklarını kapat: Sorgular başarıyla çalıştırıldıktan sonra "baglanti.Close()" metoduyla veri tabanı bağlantısı kapatılır. Bu, veri tabanı kaynaklarının serbest bırakılmasını sağlar.

Bilgi mesajı göster: "MessageBox.Show()" metoduyla kullanıcıya "Öğretmen Eklendi!" mesajı gösterilir.

TextBox ve ComboBox nesnelerini temizle: Öğretmen ekleme işleminden sonra ilgili TextBox ve ComboBox nesneleri temizlenir.

Bu kod parçası, öğretmen bilgilerini alarak veri tabanına kaydetmeyi ve ardından kullanıcıya geri bildirim vermek ve TextBox ve ComboBox nesnelerini temizlemek için gerekli işlemleri gerçekleştirir.

void DanismanIDgetir()

{

baglanti.Open();

string query = "select OgretmenID from OgretmenTable where OgretmenBolumID = '" + cxbOgrenciBolumu.Text + "' ";

SqlCommand sqlCommand = new SqlCommand(query, baglanti);

SqlDataReader reader = sqlCommand.ExecuteReader();

while (reader.Read())

{

cxbDanismanNo.Items.Add(reader.GetValue(0));

}

reader.Close();

baglanti.Close();

}

Bağlantıyı aç: "baglanti.Open()" metodu ile veri tabanı bağlantısı açılır.

Sorguyu oluştur: "string" türünde "query" adında bir değişken tanımlanır ve "OgretmenTable" tablosundan, "OgretmenBolumID" sütunu "cxbOgrenciBolumu.Text" değeriyle eşleşen öğretmenlerin "OgretmenID" sütununu seçen bir SELECT sorgusu oluşturulur.

Komut nesnesini oluştur: "SqlCommand" sınıfından "sqlCommand" adında bir komut nesnesi oluşturulur. Bu nesne, oluşturulan sorguyu ve veri tabanı bağlantısını kullanarak çalıştırır.

Verileri oku: "SqlDataReader" sınıfından "reader" adında bir nesne oluşturulur ve "sqlCommand.ExecuteReader()" metoduyla sorgudan dönen verileri okumak için kullanılır.

Verileri ComboBox'a ekle: "reader.Read()" metoduyla her bir satırı okur ve "cxbDanismanNo" adlı ComboBox'a "reader.GetValue(0)" ile elde edilen değeri ekler.

Veri tabanı kaynaklarını kapat: Veri okuma işlemi tamamlandıktan sonra "reader.Close()" metoduyla veri okuyucusu kapatılır ve "baglanti.Close()" metoduyla veri tabanı bağlantısı kapatılır.

Bu kod parçası, öğrenci ekleme panelinde seçilen bölümün danışmanlarının ID'lerini veri tabanından alarak ComboBox'a ekler. Bu sayede kullanıcı, öğrenci ekleme işlemi sırasında ilgili danışmanın ID'sini seçebilir.

## 6.5 kisiGuncellePanel.cs

private SqlConnection baglanti = new SqlConnection("Data Source=(localdb)\\ProjectModels;Initial Catalog=OgrenciOtomasyon;Integrated Security=True");

SqlConnection sınıfından baglanti adında özel bir nesne oluşturulur ve belirtilen veri tabanına bağlanmak için gerekli bağlantı dizesi kullanılır.

void OgrenciListele()

{

baglanti.Open();

SqlDataAdapter sqlDataAdapter = new SqlDataAdapter("Select \* from OgrenciTable ", baglanti);

DataTable dataTable = new DataTable();

sqlDataAdapter.Fill(dataTable);

dgwOgrenciGuncelle.DataSource = dataTable;

baglanti.Close();

}

Bağlantıyı aç: "baglanti.Open()" metodu ile veri tabanı bağlantısı açılır.

Verileri al: "SqlDataAdapter" sınıfından "sqlDataAdapter" adında bir veri adaptörü oluşturulur. Bu adaptör, "OgrenciTable" tablosundaki tüm verileri seçen bir SELECT sorgusunu çalıştırır.

Verileri bir DataTable'a doldur: "DataTable" sınıfından "dataTable" adında bir nesne oluşturulur. "sqlDataAdapter.Fill(dataTable)" metodunu kullanarak verileri alınan DataTable nesnesine doldurulur.

DataGridView'e verileri bağla: "dgwOgrenciGuncelle" adlı DataGridView kontrolünün "DataSource" özelliği, "dataTable" nesnesi olarak ayarlanır. Bu sayede DataGridView kontrolü verileri görüntüleyebilir.

Veri tabanı kaynaklarını kapat: Veri alma işlemi tamamlandıktan sonra "baglanti.Close()" metoduyla veri tabanı bağlantısı kapatılır.

Bu kod parçası, "OgrenciTable" tablosundaki tüm verileri alarak DataGridView kontrolünde görüntüler. Bu sayede kullanıcı, öğrencilerin listesini görüntüleyebilir.

void OgretmenListele()

{

baglanti.Open();

SqlDataAdapter sqlDataAdapter2 = new SqlDataAdapter("Select \* from OgretmenTable ", baglanti);

DataTable dataTable2 = new DataTable();

sqlDataAdapter2.Fill(dataTable2);

dgwOgretmenGuncelle.DataSource = dataTable2;

baglanti.Close();

}

Bağlantıyı aç: "baglanti.Open()" metodu ile veri tabanı bağlantısı açılır.

Verileri al: "SqlDataAdapter" sınıfından "sqlDataAdapter2" adında bir veri adaptörü oluşturulur. Bu adaptör, "OgretmenTable" tablosundaki tüm verileri seçen bir SELECT sorgusunu çalıştırır.

Verileri bir DataTable'a doldur: "DataTable" sınıfından "dataTable2" adında bir nesne oluşturulur. "sqlDataAdapter2.Fill(dataTable2)" metodunu kullanarak verileri alınan DataTable nesnesine doldurulur.

DataGridView'e verileri bağla: "dgwOgretmenGuncelle" adlı DataGridView kontrolünün "DataSource" özelliği, "dataTable2" nesnesi olarak ayarlanır. Bu sayede DataGridView kontrolü verileri görüntüleyebilir.

Veri tabanı kaynaklarını kapat: Veri alma işlemi tamamlandıktan sonra "baglanti.Close()" metoduyla veri tabanı bağlantısı kapatılır.

Bu kod parçası, "OgretmenTable" tablosundaki tüm verileri alarak DataGridView kontrolünde görüntüler. Bu sayede kullanıcı, öğretmenlerin listesini görüntüleyebilir.

private void button1\_Click(object sender, EventArgs e)

{

baglanti.Open();

SqlCommand cmd = new SqlCommand("update OgrenciTable set OgrenciAd = '" + tbxOgrenciAdi.Text + "',OgrenciSoyad = '" + txbOgrenciSoyadi.Text + "',OgrenciBolumID = '" + txbOgrenciBolumu.Text + "',OgrenciTC = '" + txbOgrenciTC.Text + "',OgrenciDanismanID = '" + txbOgrenciDanisman.Text + "',OgrenciSifre = '" + txbOgrenciSifre.Text + "' where OgrenciNo = '" + txbOgrenciNo.Text + "' ", baglanti);

cmd.ExecuteNonQuery();

MessageBox.Show("Öğrenci Güncellendi!");

baglanti.Close();

OgrenciListele();

}

Bağlantıyı aç: "baglanti.Open()" metoduyla veri tabanı bağlantısı açılır.

Güncelleme sorgusunu tanımla: "SqlCommand" sınıfından "cmd" adında bir komut nesnesi oluşturulur. Bu komut nesnesi, "OgrenciTable" tablosundaki belirli bir öğrencinin bilgilerini güncellemek için bir UPDATE sorgusu çalıştırır. Sorgunun parametreleri, TextBox kontrolündeki değerlerden alınır.

Güncelleme sorgusunu çalıştır: "cmd.ExecuteNonQuery()" metodu ile güncelleme sorgusu veri tabanında çalıştırılır.

Mesaj kutusu göster: "MessageBox.Show()" metodu ile kullanıcıya güncelleme işleminin tamamlandığına dair bir mesaj gösterilir.

Bağlantıyı kapat: "baglanti.Close()" metoduyla veri tabanı bağlantısı kapatılır.

Öğrenci listesini güncelle: "OgrenciListele()" metodu çağrılarak öğrenci listesinin yeniden güncellenmesi sağlanır.

Bu kod parçası, belirli bir öğrencinin bilgilerini güncellemek için veri tabanına bir UPDATE sorgusu gönderir ve ardından güncellenmiş öğrenci listesini yeniden yükler.

private void button2\_Click(object sender, EventArgs e)

{

baglanti.Open();

SqlCommand cmd = new SqlCommand("update OgretmenTable set OgretmenAd = '" + txbOgretmenAdi.Text + "',OgretmenSoyad = '" + txbOgretmenSoyadi.Text + "',OgretmenBolumID = '" + txbOgretmenBrans.Text + "',OgretmenTC = '" + txbOgretmenTC.Text + "',OgretmenSifre = '" + txbOgretmenSifre.Text + "' where OgretmenID = '" + txbOgretmenNo.Text + "' ", baglanti);

cmd.ExecuteNonQuery();

MessageBox.Show("Öğretmen Güncellendi!");

baglanti.Close();

OgretmenListele();

}

Bağlantıyı aç: "baglanti.Open()" metoduyla veri tabanı bağlantısı açılır.

Güncelleme sorgusunu tanımla: "SqlCommand" sınıfından "cmd" adında bir komut nesnesi oluşturulur. Bu komut nesnesi, "OgretmenTable" tablosundaki belirli bir öğretmenin bilgilerini güncellemek için bir UPDATE sorgusu çalıştırır. Sorgunun parametreleri, TextBox kontrolündeki değerlerden alınır.

Güncelleme sorgusunu çalıştır: "cmd.ExecuteNonQuery()" metodu ile güncelleme sorgusu veri tabanında çalıştırılır.

Mesaj kutusu göster: "MessageBox.Show()" metodu ile kullanıcıya güncelleme işleminin tamamlandığına dair bir mesaj gösterilir.

Bağlantıyı kapat: "baglanti.Close()" metoduyla veri tabanı bağlantısı kapatılır.

Öğretmen listesini güncelle: "OgretmenListele()" metodu çağrılarak öğretmen listesinin yeniden güncellenmesi sağlanır.

Bu kod parçası, belirli bir öğretmenin bilgilerini güncellemek için veri tabanına bir UPDATE sorgusu gönderir ve ardından güncellenmiş öğretmen listesini yeniden yükler.

private void dataGridView1\_CellEnter(object sender, DataGridViewCellEventArgs e)

{

tbxOgrenciAdi.Text = dgwOgrenciGuncelle.CurrentRow.Cells[1].Value.ToString();

txbOgrenciSoyadi.Text = dgwOgrenciGuncelle.CurrentRow.Cells[2].Value.ToString();

txbOgrenciTC.Text = dgwOgrenciGuncelle.CurrentRow.Cells[3].Value.ToString();

txbOgrenciBolumu.Text = dgwOgrenciGuncelle.CurrentRow.Cells[4].Value.ToString();

txbOgrenciNo.Text = dgwOgrenciGuncelle.CurrentRow.Cells[0].Value.ToString();

txbOgrenciDanisman.Text = dgwOgrenciGuncelle.CurrentRow.Cells[6].Value.ToString();

txbOgrenciSifre.Text = dgwOgrenciGuncelle.CurrentRow.Cells[7].Value.ToString();

}

Seçilen hücrenin değerlerini TextBox kontrolüne aktarır:

"dgwOgrenciGuncelle.CurrentRow.Cells[1].Value.ToString()" ifadesi, seçilen hücrenin ikinci sütunundaki değeri (öğrenci adı) alır ve "tbxOgrenciAdi" TextBox kontrolüne atar.

"dgwOgrenciGuncelle.CurrentRow.Cells[2].Value.ToString()" ifadesi, seçilen hücrenin üçüncü sütunundaki değeri (öğrenci soyadı) alır ve "txbOgrenciSoyadi" TextBox kontrolüne atar.

Benzer şekilde, diğer hücre değerleri de ilgili TextBox kontrollerine atanır.

Bu kod parçası, DataGridView'deki seçilen hücrenin değerlerini alarak, ilgili TextBox kontrollerine yerleştirir. Bu sayede, kullanıcı bir hücreye tıkladığında, o hücreye ait veriler TextBox kontrollerinde görüntülenir.

private void dataGridView2\_CellEnter(object sender, DataGridViewCellEventArgs e)

{

txbOgretmenAdi.Text = dgwOgretmenGuncelle.CurrentRow.Cells[1].Value.ToString();

txbOgretmenSoyadi.Text = dgwOgretmenGuncelle.CurrentRow.Cells[2].Value.ToString();

txbOgretmenBrans.Text = dgwOgretmenGuncelle.CurrentRow.Cells[3].Value.ToString();

txbOgretmenTC.Text = dgwOgretmenGuncelle.CurrentRow.Cells[4].Value.ToString();

txbOgretmenNo.Text = dgwOgretmenGuncelle.CurrentRow.Cells[0].Value.ToString();

txbOgretmenSifre.Text = dgwOgretmenGuncelle.CurrentRow.Cells[5].Value.ToString();

}

Seçilen hücrenin değerlerini TextBox kontrolüne aktarır:

"dgwOgretmenGuncelle.CurrentRow.Cells[1].Value.ToString()" ifadesi, seçilen hücrenin ikinci sütunundaki değeri (öğretmen adı) alır ve "txbOgretmenAdi" TextBox kontrolüne atar.

"dgwOgretmenGuncelle.CurrentRow.Cells[2].Value.ToString()" ifadesi, seçilen hücrenin üçüncü sütunundaki değeri (öğretmen soyadı) alır ve "txbOgretmenSoyadi" TextBox kontrolüne atar.

Benzer şekilde, diğer hücre değerleri de ilgili TextBox kontrollerine atanır.

Bu kod parçası, DataGridView'deki seçilen hücrenin değerlerini alarak, ilgili TextBox kontrollerine yerleştirir. Bu sayede, kullanıcı bir hücreye tıkladığında, o hücreye ait veriler TextBox kontrollerinde görüntülenir.

private void button5\_Click(object sender, EventArgs e)

{

try

{

baglanti.Open();

SqlCommand cmd = new SqlCommand("UPDATE OgrenciTable set OgrenciDonem = OgrenciDonem+1;", baglanti);

cmd.ExecuteNonQuery();

baglanti.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

Aşağıdaki adımları gerçekleştirir:

"button5\_Click" olayı tetiklendiğinde, kod çalışır.

"baglanti.Open()" ile veri tabanı bağlantısı açılır.

"SqlCommand" sınıfı kullanılarak "UPDATE OgrenciTable set OgrenciDonem = OgrenciDonem+1;" SQL sorgusu oluşturulur. Bu sorgu, "OgrenciTable" tablosunda "OgrenciDonem" sütunundaki değerleri bir artırır.

"cmd.ExecuteNonQuery()" ile sorgu veri tabanında çalıştırılır ve etkilenen satır sayısı döndürülmez (void).

"baglanti.Close()" ile veri tabanı bağlantısı kapatılır.

Hata durumunda, "catch" bloğu çalışır ve bir hata mesajı görüntülenir.

Bu kod parçası, öğrenci tablosundaki tüm öğrencilerin dönemlerini bir artırmak için bir güncelleme işlemi gerçekleştirir.

## 6.6 ogretmenPanel.cs

SqlConnection baglanti = new SqlConnection("Data Source=(localdb)\\ProjectModels;Initial Catalog=OgrenciOtomasyon;Integrated Security=True");

SqlConnection sınıfından baglanti adında özel bir nesne oluşturulur ve belirtilen veri tabanına bağlanmak için gerekli bağlantı dizesi kullanılır.

private void Form4\_Load(object sender, EventArgs e)

{

txbNo.Text = girisPanel.form\_variable;

baglanti.Open();

string query = "select \* from OgretmenTable where OgretmenID = '" + txbNo.Text + "'";

SqlCommand cmd = new SqlCommand(query, baglanti);

SqlDataReader reader = cmd.ExecuteReader();

reader.Read();

txbAdi.Text = reader.GetString(1);

txbSoyadi.Text = reader.GetString(2);

txbBrans.Text = reader.GetString(3);

reader.Close();

SqlDataAdapter sqlDataAdapter = new SqlDataAdapter("Select \* from OgrenciTable where OgrenciDanismanID = '"+ Convert.ToInt32(txbNo.Text) + "'", baglanti);

DataTable dataTable = new DataTable();

sqlDataAdapter.Fill(dataTable);

dgwOgrenciler.DataSource = dataTable;

SqlDataAdapter sqlDataAdapter2 = new SqlDataAdapter("select DersID,DersAdi from DersTable where DersID in (select OgretmenDersTable.DersID from OgretmenDersTable where OgretmenID = '"+Convert.ToInt32(txbNo.Text) +"')", baglanti);

DataTable dataTable2 = new DataTable();

sqlDataAdapter2.Fill(dataTable2);

dgwDersler.DataSource = dataTable2;

}

İşleyiş aşağıdaki adımları içerir:

"Form4\_Load" olayı tetiklendiğinde, kod çalışır.

"txbNo.Text" üzerinden öğretmenin kimlik numarası alınır. Bu, giriş panelinin "form\_variable" adlı bir değişkeninden alındı.

"baglanti.Open()" ile veri tabanı bağlantısı açılır.

Öğretmenin bilgilerini almak için "OgretmenTable" tablosunda öğretmenin kimlik numarasına göre bir sorgu oluşturulur. Bu bilgiler, öğretmenin adı, soyadı ve branşıdır.

"SqlDataReader" sınıfı kullanılarak sorgu çalıştırılır ve sonuçlar okunur. Okunan bilgiler, ilgili metin kutularına atanır.

"reader.Close()" ile veri okuyucusu kapatılır.

İlgili öğretmenin danışmanlığını yaptığı öğrencileri çekmek için "OgrenciTable" tablosunda öğrenci danışman kimlik numarasına göre bir sorgu oluşturulur. Bu sorgu "SqlDataAdapter" ve "DataTable" kullanılarak çalıştırılır ve sonuçlar "dgwOgrenciler" adlı bir DataGridView nesnesine bağlanır.

Öğretmenin verdiği dersleri çekmek için "DersTable" ve "OgretmenDersTable" tabloları arasında birleştirilmiş bir sorgu oluşturulur. Bu sorgu, öğretmenin kimlik numarasına göre verdiği dersleri seçer. Bu sorgu da "SqlDataAdapter" ve "DataTable" kullanılarak çalıştırılır ve sonuçlar "dgwDersler" adlı bir DataGridView nesnesine bağlanır.

"baglanti.Close()" ile veri tabanı bağlantısı kapatılır.

Bu kod parçası, Form4 yüklendiğinde öğretmenin bilgilerini, danışmanlık yaptığı öğrencileri ve verdiği dersleri görüntülemek için veri tabanı sorgularını kullanır.

private void button1\_Click(object sender, EventArgs e)

{

Form f = new dersOnayPanel();

f.Show();

}

private void button2\_Click(object sender, EventArgs e)

{

Form f = new notEklePanel();

f.Show();

}

"button1\_Click" olayı gerçekleştiğinde (button1 tıklandığında), "dersOnayPanel" adlı bir form oluşturulur ve gösterilir.

"button2\_Click" olayı gerçekleştiğinde (button2 tıklandığında), "notEklePanel" adlı bir form oluşturulur ve gösterilir.

Bu kod parçası, button1 tıklandığında "dersOnayPanel" formunu açar ve button2 tıklandığında "notEklePanel" formunu açar. Bu şekilde kullanıcılar, ilgili işlevleri gerçekleştirmek için bu formları kullanabilir.

## 6.7 notEklePanel.cs

SqlConnection baglanti = new SqlConnection("Data Source=(localdb)\\ProjectModels;Initial Catalog=OgrenciOtomasyon;Integrated Security=True");

SqlConnection sınıfından baglanti adında özel bir nesne oluşturulur ve belirtilen veri tabanına bağlanmak için gerekli bağlantı dizesi kullanılır.

void NotListele()

{

SqlDataAdapter sqlDataAdapter = new SqlDataAdapter("select NotID,NotTable.OgrenciDonem,Vize,Final,Butunleme,[Gecme Notu],[Gecme Durumu] from NotTable where NotTable.OgrenciDersID in (select OgrenciDersID from OgrenciDersTable where DersID in (select DersID from OgretmenDersTable where OgretmenID = '"+girisPanel.form\_variable+ "'))", baglanti);

DataTable dataTable = new DataTable();

sqlDataAdapter.Fill(dataTable);

dgwNotEkle.DataSource = dataTable;

}

Metodun çalışma mantığı şu şekildedir:

Bir SqlDataAdapter nesnesi oluşturulur ve içine SQL sorgusu verilir. Bu sorgu, belirli bir öğretmenin derslerine ait notları çekmek için birden çok alt sorguyu içerir. İç içe geçmiş sorgular kullanılarak ilgili veriler elde edilir.

Bir DataTable nesnesi oluşturulur.

SqlDataAdapter nesnesi, Fill yöntemi kullanılarak DataTable'a verileri doldurur.

DataTable, "dgwNotEkle" DataGridView nesnesine bağlanır ve notları görüntülemek için kullanılır.

Bu şekilde, "NotListele" metodu çağrıldığında ilgili öğretmenin derslerine ait notlar, "dgwNotEkle" DataGridView kontrolünde görüntülenir.

private void button2\_Click(object sender, EventArgs e)

{

try

{

baglanti.Open();

SqlCommand cmd = new SqlCommand("update NotTable set Vize = '" + Convert.ToInt32(txbVize.Text) + "',Final = '" + Convert.ToInt32(txbFinal.Text) + "',Butunleme = '" + Convert.ToInt32(txbBut.Text) + "' where NotID = ('" + dgwNotEkle.CurrentRow.Cells[0].Value + "') ", baglanti);

cmd.ExecuteNonQuery();

baglanti.Close();

if (Convert.ToInt32(dgwNotEkle.CurrentRow.Cells[4].Value) == 0)

{

baglanti.Open();

SqlCommand cmd2 = new SqlCommand("update NotTable set [Gecme Notu] = '"+((Convert.ToInt32(txbVize.Text)\*0.4)+(Convert.ToInt32(txbFinal.Text) \* 0.6)) + "' where NotID = ('" + dgwNotEkle.CurrentRow.Cells[0].Value + "') ", baglanti);

cmd2.ExecuteNonQuery();

baglanti.Close();

Gectimi();

NotListele();

}

else if (Convert.ToInt32(dgwNotEkle.CurrentRow.Cells[4].Value) != 0)

{

baglanti.Open();

SqlCommand cmd3 = new SqlCommand("update NotTable set [Gecme Notu] = '" + ((Convert.ToInt32(txbVize.Text) \* 0.4) + (Convert.ToInt32(txbBut.Text) \* 0.6)) + "' where NotID = ('" + dgwNotEkle.CurrentRow.Cells[0].Value + "')", baglanti);

cmd3.ExecuteNonQuery();

baglanti.Close();

Gectimi();

NotListele();

}

baglanti.Close();

Gectimi();

NotListele();

txbBut.Clear();

txbFinal.Clear();

txbVize.Clear();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

baglanti.Close();

}

}

Kodun çalışma mantığı şu şekildedir:

Öncelikle veri tabanı bağlantısı açılır.

İlk SqlCommand nesnesiyle "NotTable" tablosundaki belirli bir notun Vize, Final ve Bütünleme değerleri güncellenir. Güncelleme işlemi, SqlCommand nesnesinin ExecuteNonQuery yöntemiyle gerçekleştirilir.

Ardından, if-else if bloğu kullanılarak notun Bütünleme durumuna göre [Gecme Notu] değeri hesaplanır ve güncellenir. Hesaplama, Vize ve Final veya Vize ve Bütünleme notlarının ağırlıklı ortalamasının alınmasıyla yapılır.

Veri tabanı bağlantısı kapatılır.

Gectimi() metodu çağrılarak öğrencinin geçme durumu güncellenir.

NotListele() metodu çağrılarak notlar yeniden yüklenir ve DataGridView kontrolünde görüntülenir.

TextBox kontrollerinin içeriği temizlenir.

Eğer herhangi bir hata oluşursa, catch bloğu çalışır ve hata mesajı gösterilir. Veri tabanı bağlantısı kapatılır.

Bu kod parçası, belirli bir notun Vize, Final, Bütünleme ve [Gecme Notu] değerlerini güncellemek için kullanılır. Ayrıca, geçme durumunu güncellemek ve notları yeniden yüklemek için ilgili metotları çağırır.

void Gectimi()

{

baglanti.Open();

SqlCommand cmd = new SqlCommand("update NotTable set [Gecme Durumu] = 'Basarili' where NotTable.[Gecme Notu] >= 60 and NotID ='"+ dgwNotEkle.CurrentRow.Cells[0].Value + "'", baglanti);

SqlCommand cmd2 = new SqlCommand("update NotTable set [Gecme Durumu] = 'Basarisiz' where NotTable.[Gecme Notu] < 60 and NotID = '" + dgwNotEkle.CurrentRow.Cells[0].Value + "' ", baglanti);

cmd2.ExecuteNonQuery();

cmd.ExecuteNonQuery();

baglanti.Close();

}

Kodun çalışma mantığı şu şekildedir:

Öncelikle veri tabanı bağlantısı açılır.

İki ayrı SqlCommand nesnesi tanımlanır. İlk SqlCommand nesnesi, [Gecme Notu] değeri 60 veya daha büyük olan notların [Gecme Durumu] sütununu "Basarili" olarak günceller. İkinci SqlCommand nesnesi ise [Gecme Notu] değeri 60'dan küçük olan notların [Gecme Durumu] sütununu "Basarisiz" olarak günceller.

Her iki güncelleme işlemi de SqlCommand nesnesinin ExecuteNonQuery yöntemiyle gerçekleştirilir.

Veri tabanı bağlantısı kapatılır.

Bu metot, belirli bir notun [Gecme Notu] değerine göre [Gecme Durumu] sütununu güncellemek için kullanılır. [Gecme Notu] değeri 60 veya daha büyükse "Basarili" olarak güncellenir, değer 60'dan küçükse "Basarisiz" olarak güncellenir.

private void dataGridView1\_CellEnter(object sender, DataGridViewCellEventArgs e)

{

txbVize.Text = dgwNotEkle.CurrentRow.Cells[2].Value.ToString();

txbFinal.Text = dgwNotEkle.CurrentRow.Cells[3].Value.ToString();

txbBut.Text = dgwNotEkle.CurrentRow.Cells[4].Value.ToString();

}

Kodun çalışma mantığı şu şekildedir:

Bir DataGridView hücresine girildiğinde, olay işleyicisi tetiklenir.

Olay işleyicisi içinde, txbVize, txbFinal ve txbBut adlı TextBox kontrollerine, seçilen hücrenin ilgili sütunlarındaki değerler aktarılır.

txbVize.Text = dgwNotEkle.CurrentRow.Cells[2].Value.ToString(); Seçilen hücrenin 2. sütunundaki değer, txbVize TextBox'ına aktarılır.

txbFinal.Text = dgwNotEkle.CurrentRow.Cells[3].Value.ToString(); Seçilen hücrenin 3. sütunundaki değer, txbFinal TextBox'ına aktarılır.

txbBut.Text = dgwNotEkle.CurrentRow.Cells[4].Value.ToString(); Seçilen hücrenin 4. sütunundaki değer, txbBut TextBox'ına aktarılır.

Bu kod parçası, DataGridView'deki seçili hücrenin değerlerini TextBox kontrollerine aktarmak için kullanılır. Bu sayede, kullanıcı seçtiği hücrenin değerini TextBox'lar aracılığıyla görüntüleyebilir veya düzenleyebilir.

## 6.8 dersOnayPanel.cs

SqlConnection baglanti = new SqlConnection("Data Source=(localdb)\\ProjectModels;Initial Catalog=OgrenciOtomasyon;Integrated Security=True");

SqlConnection sınıfından baglanti adında özel bir nesne oluşturulur ve belirtilen veri tabanına bağlanmak için gerekli bağlantı dizesi kullanılır.

void DanismanOnayListele()

{

SqlDataAdapter sqlDataAdapter = new SqlDataAdapter("select OgrenciDersID,OgrenciTable.OgrenciNo,concat(OgrenciTable.OgrenciAd,' ',OgrenciTable.OgrenciSoyad) as Ogrenci,DanismanOnay,concat(OgretmenAd,' ',OgretmenSoyad) as Danisman,DersAdi,OgrenciDonem from OgrenciTable,OgrenciDersTable,OgretmenTable,DersTable where OgrenciTable.OgrenciDanismanID = '" + girisPanel.form\_variable + "' and OgrenciDersTable.OgrenciNo = OgrenciTable.OgrenciNo and OgrenciDersTable.DanismanOnay like ' Onay%' and OgretmenAd = (select OgretmenAd from OgretmenTable where OgretmenID = '" + girisPanel.form\_variable+"') and OgrenciDersTable.DersID = DersTable.DersID ;", baglanti);

DataTable dataTable = new DataTable();

sqlDataAdapter.Fill(dataTable);

dataGridView1.DataSource = dataTable;

}

Kodun çalışma mantığı şu şekildedir:

Bir SqlDataAdapter oluşturulur ve bir SQL sorgusu kullanarak veriler alınır. SQL sorgusu, öğrenci, öğretmen, ders ve öğrenci ders tablolarını ilişkilendirir ve belirli bir danışmanın onaylanmış öğrenci derslerini seçer. Sorguda kullanılan filtreler "OgrenciTable.OgrenciDanismanID", "OgrenciDersTable.DanismanOnay" ve "OgretmenAd" değerlerine dayanır.

Bir DataTable oluşturulur ve SqlDataAdapter ile doldurulur. Bu DataTable, seçilen verileri depolamak için kullanılır.

DataGridView kontrolüne DataTable bağlanır ve veriler görüntülenir.

Bu kod parçası, belirli bir danışmanın onaylanmış öğrenci derslerini DataGridView üzerinde listelemek için kullanılır. Bu sayede danışman, onaylanmış dersleri görebilir ve gerekli işlemleri yapabilir.

private void button1\_Click(object sender, EventArgs e)

{

try

{

baglanti.Open();

if (dataGridView1.CurrentRow != null && dataGridView1.CurrentRow.Cells[0].Value != null && dataGridView1.CurrentRow.Cells[6].Value != null)

{

string sql = "insert into NotTable (OgrenciDersID, OgrenciDonem) values (@OgrenciDersID, @OgrenciDonem)";

SqlCommand sqlCommand = new SqlCommand(sql, baglanti);

sqlCommand.Parameters.AddWithValue("@OgrenciDersID", Convert.ToInt32(dataGridView1.CurrentRow.Cells[0].Value));

sqlCommand.Parameters.AddWithValue("@OgrenciDonem", Convert.ToInt32(dataGridView1.CurrentRow.Cells[6].Value));

sqlCommand.ExecuteNonQuery();

SqlCommand cmd = new SqlCommand("update OgrenciDersTable set DanismanOnay = 'Kesin Kayıt' where OgrenciDersID = @OgrenciDersID", baglanti);

cmd.Parameters.AddWithValue("@OgrenciDersID", Convert.ToInt32(dataGridView1.CurrentRow.Cells[0].Value));

cmd.ExecuteNonQuery();

}

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

finally

{

baglanti.Close();

DanismanOnayListele();

}

}

Kodun çalışma mantığı şu şekildedir:

Bağlantı açılır.

Koşul ifadesi kullanılarak, DataGridView'de seçili bir satırın ve ilgili hücrelerin değerlerinin boş olmadığı kontrol edilir.

Eğer koşul sağlanıyorsa, bir SQL sorgusu kullanılarak "NotTable" tablosuna yeni bir kayıt eklenir. SQL sorgusu, "OgrenciDersID" ve "OgrenciDonem" sütunlarına parametreler kullanarak değerleri ekler. Parametreler, SqlCommand nesnesi ile eklenir ve hücre değerleri dönüştürülerek atanır.

Bir SqlCommand nesnesi daha oluşturulur ve "OgrenciDersTable" tablosundaki ilgili kaydın "DanismanOnay" sütunu "Kesin Kayıt" olarak güncellenir. Bu işlem için de "OgrenciDersID" parametresi kullanılır.

Hata durumunda MessageBox ile bir hata mesajı gösterilir.

Bağlantı kapatılır.

"DanismanOnayListele" metodunu çağırarak DataGridView'i günceller.

Bu kod parçası, DataGridView'de seçili bir öğrenci ders kaydını "NotTable" tablosuna ekleyerek ve "OgrenciDersTable" tablosundaki ilgili kaydı güncelleyerek, danışmanın onayladığı bir ders kaydının kesin kaydını yapmayı sağlar. Ayrıca, işlemden sonra DataGridView'deki verileri günceller.

## 6.9 ogrenciPanel.cs

SqlConnection baglanti = new SqlConnection("Data Source=(localdb)\\ProjectModels;Initial Catalog=OgrenciOtomasyon;Integrated Security=True");

SqlConnection sınıfından baglanti adında özel bir nesne oluşturulur ve belirtilen veri tabanına bağlanmak için gerekli bağlantı dizesi kullanılır.

private void Form2\_Load(object sender, EventArgs e)

{

txbNo.Text = girisPanel.form\_variable;

baglanti.Open();

string query = "select \* from OgrenciTable where OgrenciNo = '"+txbNo.Text+"'";

SqlCommand cmd = new SqlCommand(query,baglanti);

SqlDataReader reader = cmd.ExecuteReader();

reader.Read();

txbAdi.Text = reader.GetString(1);

txbSoyadi.Text = reader.GetString(2);

txbBolumu.Text = reader.GetString(4);

reader.Close();

SqlDataAdapter sqlDataAdapter = new SqlDataAdapter("Select \* from DersTable", baglanti);

DataTable dataTable = new DataTable();

sqlDataAdapter.Fill(dataTable);

dgwdDersler.DataSource = dataTable;

baglanti.Close();

}

Kodun çalışma mantığı şu şekildedir:

"txbNo" adlı TextBox'a "girisPanel" formundan alınan "form\_variable" değeri atanır. Bu değer, giriş yapan öğrencinin numarasını temsil eder.

Bağlantı açılır.

Bir SQL sorgusu kullanılarak "OgrenciTable" tablosundan "OgrenciNo" değeri "txbNo.Text" ile eşleşen bir kayıt seçilir.

SqlCommand nesnesi oluşturulur ve sorgu ile bağlantı ilişkilendirilir. SqlDataReader kullanarak sorgu sonucu okunur.

Okuma işlemi gerçekleştiğinde, SqlDataReader üzerinde "Read" metodu çağırılarak bir sonraki kayda geçilir.

Reader nesnesi üzerinden ilgili hücrelerin değerleri alınarak "txbAdi", "txbSoyadi" ve "txbBolumu" TextBox kontrollerine atanır.

Reader kapatılır.

SqlDataAdapter kullanılarak "DersTable" tablosundan tüm veriler alınır ve bir DataTable nesnesine aktarılır.

DataTable, DataGridView olan "dgwdDersler" kontrolünün veri kaynağı olarak atanır.

Bağlantı kapatılır.

Bu kod parçası, Form2'nin yüklenme olayında öğrenci bilgilerini ve tüm dersleri DataGridView'de görüntüler.

private void dataGridView1\_CellEnter\_1(object sender, DataGridViewCellEventArgs e)

{

txbDersAdi.Text = dgwdDersler.CurrentRow.Cells[1].Value.ToString();

txbDersKredisi.Text = dgwdDersler.CurrentRow.Cells[2].Value.ToString();

txbDersAKTS.Text = dgwdDersler.CurrentRow.Cells[3].Value.ToString();

txbDersNo.Text = dgwdDersler.CurrentRow.Cells[0].Value.ToString();

}

Kodun çalışma mantığı şu şekildedir:

"txbDersAdi", "txbDersKredisi", "txbDersAKTS" ve "txbDersNo" adlı TextBox kontrollerine, "dgwdDersler" DataGridView kontrolünün seçilen satırının ilgili hücrelerinin değerleri atanır.

"dgwdDersler.CurrentRow.Cells[1].Value.ToString()" ifadesi, seçilen satırın ikinci hücresinin değerini "txbDersAdi" TextBox'ına atar.

Benzer şekilde diğer TextBox kontrollerine ilgili hücrelerin değerleri atanır.

Bu kod parçası, DataGridView'deki seçili dersin bilgilerini TextBox kontrollerine yükler. Bu şekilde kullanıcı, DataGridView üzerindeki dersleri seçtiğinde ders bilgilerini TextBox kontrollerinde görüntüleyebilir.

private void button1\_Click(object sender, EventArgs e)

{

baglanti.Open();

try

{

string query = "Insert into OgrenciDersTable (OgrenciNo,DersID) values('" + txbNo.Text + "','" + txbDersNo.Text + "')";

SqlCommand sqlCommand = new SqlCommand(query, baglanti);

sqlCommand.ExecuteNonQuery();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

finally

{

MessageBox.Show("Ders Listeye Eklendi!");

baglanti.Close();

}

}

Kodun çalışma mantığı şu şekildedir:

Veri tabanı bağlantısı açılır.

Bir "query" değişkeni oluşturulur ve "OgrenciDersTable" adlı tabloya öğrenci numarası ve ders ID'si eklemek için kullanılan INSERT INTO sorgusu tanımlanır. Sorgunun değerleri, "txbNo.Text" (öğrenci numarası) ve "txbDersNo.Text" (ders ID'si) TextBox'larından alınır.

Bir SqlCommand nesnesi oluşturulur ve sorgu ile bağlantı ilişkilendirilir.

"ExecuteNonQuery" yöntemi kullanılarak sorgu veri tabanında çalıştırılır ve yeni kayıt eklenir.

Hata durumunda, bir hata mesajı görüntülenir.

"finally" bloğunda, bir bilgi mesajı ("Ders Listeye Eklendi!") görüntülenir ve veri tabanı bağlantısı kapatılır.

Bu kod parçası, öğrenci ders listesine yeni bir ders eklemek için kullanılır. Öğrenci numarası ve ders ID'si TextBox'larındaki değerler kullanılarak "OgrenciDersTable" tablosuna yeni bir kayıt eklenir.

private void button2\_Click(object sender, EventArgs e)

{

Form f = new dersSecimPanel();

f.Show();

}

private void button3\_Click(object sender, EventArgs e)

{

Form f = new notGoruntulePanel();

f.Show();

}

"button2\_Click" olay işleyicisi, bir düğmeye tıklandığında çalışır. Bu olay işleyicisi, "dersSecimPanel" adlı bir formun görüntülenmesini sağlar.

İlk olarak, "dersSecimPanel" adlı bir form örneği oluşturulur.

Oluşturulan form örneği "f" kullanılarak "Show" yöntemi çağrılır ve form görüntülenir.

"button3\_Click" olay işleyicisi, bir düğmeye tıklandığında çalışır. Bu olay işleyicisi, "notGoruntulePanel" adlı bir formun görüntülenmesini sağlar.

İlk olarak, "notGoruntulePanel" adlı bir form örneği oluşturulur.

Oluşturulan form örneği "f" kullanılarak "Show" yöntemi çağrılır ve form görüntülenir.

Bu kod parçası, iki farklı formun görüntülenmesi için düğmelere tıklama olaylarını tanımlar.

## 6.10 notGoruntulePanel.cs

SqlConnection baglanti = new SqlConnection("Data Source=(localdb)\\ProjectModels;Initial Catalog=OgrenciOtomasyon;Integrated Security=True");

SqlConnection sınıfından baglanti adında özel bir nesne oluşturulur ve belirtilen veri tabanına bağlanmak için gerekli bağlantı dizesi kullanılır.

private void notGoruntulemePanel(object sender, EventArgs e)

{

baglanti.Open();

string query = "select OgrenciDersID,Vize,Final,Butunleme,[Gecme Notu],[Gecme Durumu],OgrenciDonem from NotTable where NotTable.OgrenciDersID in (select OgrenciDersID from OgrenciDersTable where OgrenciNo ='" + Convert.ToInt32(girisPanel.form\_variable) + "' ) ";

SqlDataAdapter sqlDataAdapter = new SqlDataAdapter(query, baglanti);

DataTable dataTable = new DataTable();

sqlDataAdapter.Fill(dataTable);

dgwNot.DataSource = dataTable;

baglanti.Close();

}

İlk olarak, veri tabanı bağlantısı açılır.

Bir SQL sorgusu oluşturulur. Bu sorgu, "NotTable" tablosundan belirli koşullara sahip verileri seçer. Koşullar, "OgrenciDersID" sütununun "OgrenciDersTable" tablosunda belirli bir "OgrenciNo" değeriyle eşleşmesidir. "girisPanel.form\_variable" değişkeni, giriş panelindeki formdan alınan değeri temsil eder.

Oluşturulan SQL sorgusu ve bağlantı kullanılarak bir SqlDataAdapter oluşturulur.

Bir DataTable oluşturulur.

SqlDataAdapter, SQL sorgusundan elde edilen verileri DataTable'a doldurur.

DataTable, DataGridView kontrolü olan "dgwNot" nesnesine atanır ve verilerin görüntülenmesi sağlanır.

Veri tabanı bağlantısı kapatılır.

Bu kod parçası, " notGoruntulemePanel " adlı formun yüklenmesi sırasında "NotTable" tablosundan belirli verileri alarak DataGridView kontrolünde görüntülemeyi sağlar.

private void dataGridView1\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

txbDersID.Text = dgwNot.CurrentRow.Cells[0].Value.ToString();

}

Olay işleyicisi, tıklanan hücrenin satır ve sütun bilgilerini içeren "DataGridViewCellEventArgs" parametresini alır.

"txbDersID" adlı bir TextBox kontrolüne, tıklanan hücrenin satırındaki 0. sütunun değeri atanır. Bu değer, "dgwNot" DataGridView'inin "CurrentRow" özelliği aracılığıyla ve ardından tıklanan hücrenin 0. sütununun değeri aracılığıyla elde edilir.

Bu kod parçası, bir hücre içeriği tıklandığında tıklanan hücrenin değerini "txbDersID" TextBox kontrolüne atar. Bu, kullanıcının tıkladığı hücrenin değerini bir başka kontrolde kullanmak için kullanılabilir.

private void textBox1\_TextChanged(object sender, EventArgs e)

{

baglanti.Close();

baglanti.Open();

string query = "select DersID from OgrenciDersTable where OgrenciDersID = '" + Convert.ToInt32(dgwNot.CurrentRow.Cells[0].Value.ToString()) + "'";

SqlCommand cmd = new SqlCommand(query, baglanti);

SqlDataReader reader = cmd.ExecuteReader();

reader.Read();

txbDersID.Text = Convert.ToString(reader.GetValue(0));

reader.Close();

baglanti.Close();

baglanti.Open();

string query2 = "select DersAdi from DersTable where DersID = '" + Convert.ToInt32(txbDersID.Text) + "'";

SqlCommand cmd2 = new SqlCommand(query2, baglanti);

SqlDataReader reader2 = cmd2.ExecuteReader();

reader2.Read();

txbDersAdi.Text = Convert.ToString(reader2.GetValue(0));

reader2.Close();

baglanti.Close();

}

İlk olarak, mevcut bağlantıyı kapatır ve yeniden açar.

Ardından, "dgwNot" DataGridView kontrolünde seçili satırın 0. hücresinin değerini alır ve bunu "OgrenciDersID" olarak kullanır.

Bu "OgrenciDersID" değeri, "OgrenciDersTable" tablosunda "DersID" sütununu sorgular. Sorgu sonucunu okuyarak elde edilen "DersID" değeri, "txbDersID" TextBox kontrolüne atanır.

Bağlantı kapatılır ve tekrar açılır.

Şimdi, "txbDersID" TextBox kontrolündeki "DersID" değerini kullanarak "DersTable" tablosunda ilgili "DersAdi" sütununu sorgular. Sorgu sonucunu okuyarak elde edilen "DersAdi" değeri, "txbDersAdi" TextBox kontrolüne atanır.

Bu kod parçası, "textBox1" TextBox kontrolünde metin değiştiğinde, ilgili veri tabanı tablolarını sorgulayarak "txbDersID" ve "txbDersAdi" TextBox kontrollerine ilgili değerleri atar. Bu sayede, kullanıcının "textBox1" TextBox'ına girilen metine bağlı olarak diğer kontrollerde ilgili verileri güncelleyebilirsiniz.

private void dataGridView1\_CellEnter(object sender, DataGridViewCellEventArgs e)

{

txbDersID.Text = dgwNot.CurrentRow.Cells[0].Value.ToString();

}

Olay işleyicisi başlatıldığında, "dgwNot" DataGridView kontrolünde bulunan mevcut satırın 0. hücresinin değerini alır.

Bu hücre değeri, bir metin olarak "txbDersID" TextBox kontrolüne atanır.

Yani, kullanıcı "dataGridView1" DataGridView kontrolünde bir hücreye girdiğinde veya hücreye odaklandığında, bu olay işleyicisi tetiklenir ve seçilen hücrenin değeri "txbDersID" TextBox kontrolüne yazılır. Bu sayede, kullanıcı seçtiği hücrenin değerini "txbDersID" TextBox'ında görebilir veya bu değeri başka bir amaçla kullanabilirsiniz.

## 6.11 dersSecimPanel.cs

SqlConnection baglanti = new SqlConnection("Data Source=(localdb)\\ProjectModels;Initial Catalog=OgrenciOtomasyon;Integrated Security=True");

SqlConnection sınıfından baglanti adında özel bir nesne oluşturulur ve belirtilen veri tabanına bağlanmak için gerekli bağlantı dizesi kullanılır.

void DersKayıtListele()

{

string query = "select \* from OgrenciDersTable where OgrenciNo = @OgrenciNo";

SqlDataAdapter sqlDataAdapter = new SqlDataAdapter(query, baglanti);

sqlDataAdapter.SelectCommand.Parameters.AddWithValue("@OgrenciNo", Convert.ToInt32(girisPanel.form\_variable));

DataTable dataTable = new DataTable();

sqlDataAdapter.Fill(dataTable);

dgwEkliDersler.DataSource = dataTable;

}

İşlev başladığında, "OgrenciNo" parametresini kullanarak bir sorgu oluşturulur. Sorgu, "OgrenciDersTable" tablosundan öğrenci numarasına göre kayıtlı dersleri seçer.

Bir SqlDataAdapter nesnesi oluşturulur ve sorgu ile bağlantıyı kullanarak verileri alır. "OgrenciNo" parametresi, SqlCommand'un Parameters koleksiyonuna eklenir.

Bir DataTable oluşturulur ve SqlDataAdapter aracılığıyla veriler bu tabloya doldurulur.

DataTable, "dgwEkliDersler" DataGridView kontrolünün veri kaynağı olarak atanır. Bu sayede, kaydedilmiş dersler "dgwEkliDersler" kontrolünde görüntülenir.

Bu işlev, öğrencinin kayıtlı derslerini getirerek kullanıcıya görüntüleme imkanı sağlar.

private void button2\_Click(object sender, EventArgs e)

{

try

{

baglanti.Open();

if (dgwEkliDersler.CurrentRow != null && dgwEkliDersler.CurrentRow.Cells[0].Value != null)

{

SqlCommand cmd = new SqlCommand("delete from OgrenciDersTable where OgrenciDersID = @OgrenciDersID", baglanti);

cmd.Parameters.AddWithValue("@OgrenciDersID", Convert.ToInt32(dgwEkliDersler.CurrentRow.Cells[0].Value));

cmd.ExecuteNonQuery();

}

}

catch (Exception ex)

{

MessageBox.Show("Hata oluştu: " + ex.Message);

}

finally

{

baglanti.Close();

DersKayıtListele();

}

}

İşlev başladığında, bağlantı açılır.

Ardından, seçilen satırın mevcut olduğu ve en az bir hücre değeri içerdiği kontrol edilir. Eğer bu koşullar sağlanıyorsa, SqlCommand nesnesi oluşturulur ve veri tabanından kaydı silmek için bir sorgu çalıştırılır. Sorgu, "OgrenciDersID" parametresini kullanır ve bu parametre, seçilen satırdaki değere göre ayarlanır.

SqlCommand nesnesinin ExecuteNonQuery() yöntemi çağrılarak sorgu çalıştırılır ve kayıt silinir.

Hata durumunda bir istisna (Exception) oluşursa, kullanıcıya hata mesajı gösterilir.

Son olarak, bağlantı kapatılır ve "DersKayıtListele" işlevi çağrılarak güncellenmiş ders kaydı listesi alınır ve gösterilir.

Bu işlev, kullanıcının "dgwEkliDersler" kontrolünden seçtiği bir ders kaydını silmesini sağlar.

private void button1\_Click(object sender, EventArgs e)

{

try

{

baglanti.Open();

if (dgwEkliDersler.CurrentRow != null && dgwEkliDersler.CurrentRow.Cells[0].Value != null)

{

SqlCommand cmd = new SqlCommand("update OgrenciDersTable set DanismanOnay = @DanismanOnay where OgrenciDersID = @OgrenciDersID", baglanti);

cmd.Parameters.AddWithValue("@DanismanOnay", "Onay Bekleniyor");

cmd.Parameters.AddWithValue("@OgrenciDersID", Convert.ToInt32(dgwEkliDersler.CurrentRow.Cells[0].Value));

cmd.ExecuteNonQuery();

}

}

catch (Exception ex)

{

MessageBox.Show("Hata oluştu: " + ex.Message);

}

finally

{

baglanti.Close();

DersKayıtListele();

}

}

İşlev başladığında, bağlantı açılır.

Ardından, seçilen satırın mevcut olduğu ve en az bir hücre değeri içerdiği kontrol edilir.

Eğer bu koşullar sağlanıyorsa, SqlCommand nesnesi oluşturulur ve veri tabanındaki kaydı güncellemek için bir sorgu çalıştırılır. Sorgu, "DanismanOnay" ve "OgrenciDersID" parametrelerini kullanır. "DanismanOnay" parametresi "Onay Bekleniyor" olarak ayarlanır ve "OgrenciDersID" parametresi seçilen satırdaki değere göre ayarlanır.

SqlCommand nesnesinin ExecuteNonQuery() yöntemi çağrılarak sorgu çalıştırılır ve kayıt güncellenir.

Hata durumunda bir istisna (Exception) oluşursa, kullanıcıya hata mesajı gösterilir.

Son olarak, bağlantı kapatılır ve "DersKayıtListele" işlevi çağrılarak güncellenmiş ders kaydı listesi alınır ve gösterilir.

Bu işlev, kullanıcının "dgwEkliDersler" kontrolünden seçtiği bir ders kaydının "DanismanOnay" alanını "Onay Bekleniyor" olarak güncellemesini sağlar.

## 6.12 User.cs

class User

{

private string userName;

private string password;

public void setUserName(string uName)

{

userName = uName;

}

public string getUserName()

{

return userName;

}

public void setPassword(string pwd)

{

password = pwd;

}

public string getPassword()

{

return password;

}

}

Sınıfın özellikleri şunlardır:

userName: Kullanıcı adını temsil eden özel bir dize alanı.

password: Şifreyi temsil eden özel bir dize alanı.

Ayrıca, sınıf aşağıdaki yöntemleri içerir:

setUserName(string uName): Kullanıcı adını ayarlamak için kullanılan bir yöntem. Bu yöntem, parametre olarak bir dize alır ve userName alanına atanır.

getUserName(): Kullanıcı adını döndüren bir yöntem. userName alanının değerini döndürür.

setPassword(string pwd): Şifreyi ayarlamak için kullanılan bir yöntem. Bu yöntem, parametre olarak bir dize alır ve password alanına atanır.

getPassword(): Şifreyi döndüren bir yöntem. password alanının değerini döndürür.

Bu sınıf, bir kullanıcının kullanıcı adı ve şifre bilgilerini tutmak için kullanılabilir. Kullanıcı adı ve şifre değerlerine get ve set yöntemleri aracılığıyla erişebilirsiniz.

7. SONSÖZ

Bu tez çalışması, C#, MSSQL, Visual Studio ve Windows Form kullanarak tasarlanmış küçük çaplı bir öğrenci otomasyonu projesini kapsamaktadır. Bu çalışma, üniversite öğrencileri için önemli bir ihtiyaca cevap vermek amacıyla gerçekleştirilmiştir.

Otomasyon sistemimizde, giriş, şifre ve kullanıcı yönetimi, admin yani memur ekranı, öğretmen ekranı, not görüntüleme ekranı, not giriş ekranı, öğretmen ve öğrenci ekleme-güncelleme ekranı, danışman ders onay ekranı, ders seçim ekranı gibi birçok form bulunmaktadır. Ayrıca, bu formların işlevselliğini desteklemek için öğrenci, öğretmen, memur, öğretmen dersleri, öğrenci dersleri, kullanıcı, ders, bölüm ve not tabloları oluşturulmuştur.

Tez çalışması, birçok aşamayı içeren bir sürecin sonucunda ortaya çıkmıştır. Yazılım geliştirme sürecinde kullanılan teknolojilerin yanı sıra analiz, tasarım, kodlama ve test gibi adımlar da dikkate alınmıştır. Bu çalışmada, öğrenci otomasyonu projesinin başarılı bir şekilde tamamlanması için yoğun bir çaba sarf edilmiştir.

Bu çalışmada elde edilen deneyimler ve kazanılan bilgiler, yazılım geliştirme alanında ilerlemek isteyen diğer araştırmacılar ve geliştiriciler için faydalı olabilecektir. Bu tez çalışması, yazılım projelerinin planlanması, tasarlanması ve uygulanması sürecinde rehberlik edebilecek bir kaynak olarak değerlendirilebilir.

Son olarak, bu tez çalışmasının tamamlanması benim için büyük bir memnuniyet kaynağıdır. Bu projede edindiğim deneyimler ve kazanılan bilgiler, kariyerimin ilerleyen aşamalarında beni daha da güçlendirecektir.

8 . KAYNAKÇA

1. Sektörüm Dergisi. (Erişim tarihi: 8 Haziran 2023). Otomasyon Nedir? Otomasyon Sistemleri Ne İşe Yarar ve Nerelerde Kullanılır?, <https://www.sektorumdergisi.com/otomasyon-nedir-otomasyon-sistemleri-ne-ise-yarar-ve-nerelerde-kullanilir/>
2. Wikipedia. (Erişim tarihi: 8 Haziran 2023). C#. <https://tr.wikipedia.org/wiki/C_Sharp>
3. Wikipedia. (Erişim tarihi: 8 Haziran 2023). Nesne yönelimli programlama. <https://tr.wikipedia.org/wiki/Nesne_y%C3%B6nelimli_programlama>
4. Wikipedia. (Erişim tarihi: 8 Haziran 2023). Windows Forms. <https://en.wikipedia.org/wiki/Windows_Forms>
5. Oracle. (Erişim tarihi: 8 Haziran 2023). What Is a Database? <https://www.oracle.com/tr/database/what-is-database/>
6. Gtech. (Erişim tarihi: 8 Haziran 2023). Microsoft SQL Server Nedir, Ne İçin Kullanılır? <https://www.gtech.com.tr/microsoft-sql-server-nedir-ne-icin-kullanilir/>